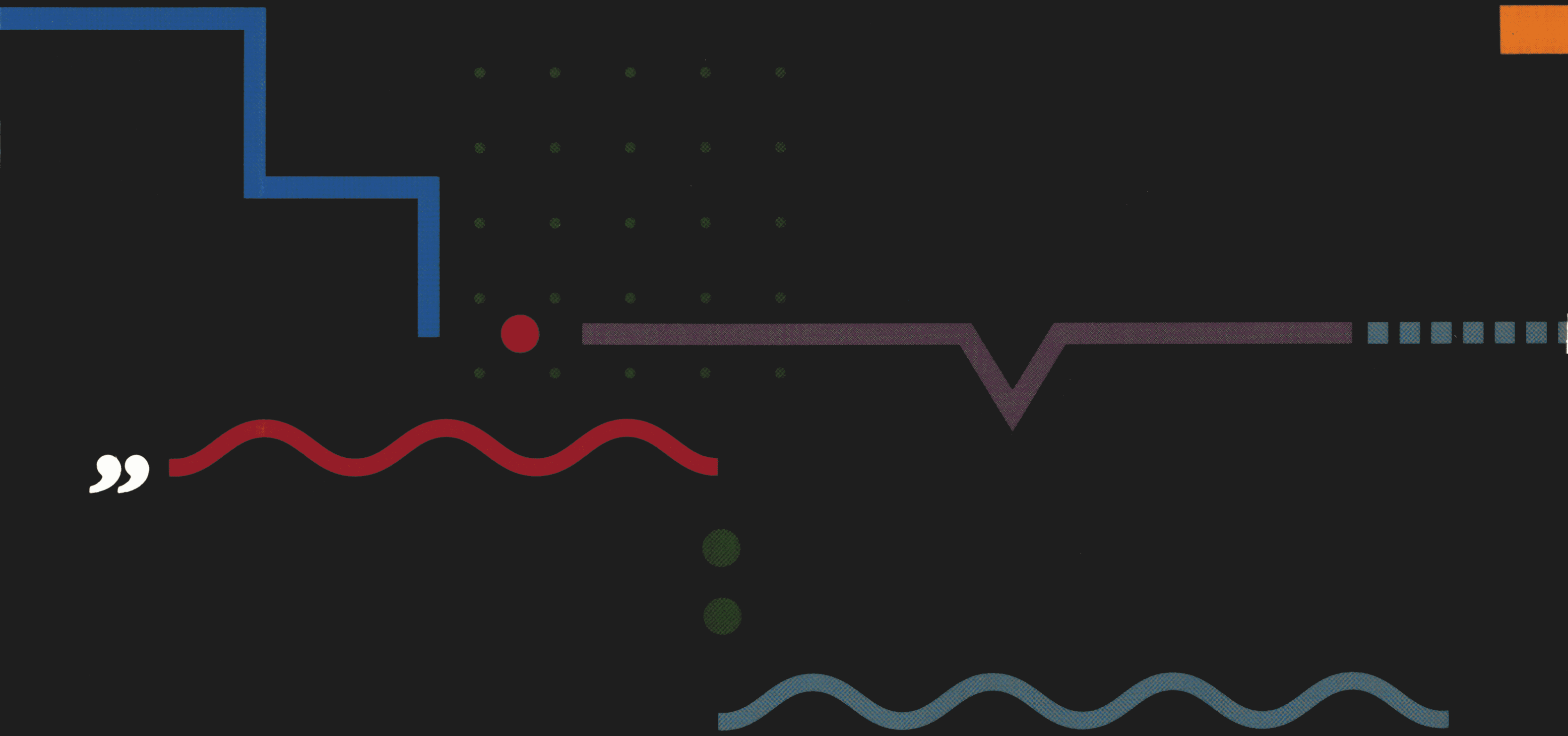
AMIGA® ROM Kernel Reference Manual

INCLUDES AND AUTODOCS



AMIGA TECHNICAL REFERENCE SERIES

COMMODORE-AMIGA, INC.

THIRD EDITION

AMIGA®

ROM Kernel Reference Manual: Includes and Autodocs

Third Edition

Commodore-Amiga, Incorporated

AMIGA TECHNICAL REFERENCE SERIES

Addison-Wesley Publishing Company, Inc.

Reading, Massachusetts Menlo Park, California New York
Don Mills, Ontario Wokingham, England Amsterdam Bonn
Sydney Singapore Tokyo Madrid San Juan
Paris Seoul Milan Mexico City Taipei

Contributors:
Steve Beats, Dave Berezowski, Ray Brand, Robert Burns, Peter Cherna, Eric Cotton, Sam Dicker, Andy Finkel, Darren Greenwald, William Hawes, Larry Hildebrand, Randell Jesup, Neil Katin, Kevin Klop, Bill Koester, Dale Larson, Dale Luck, Jim Mackraz, R.J. Mical, Bryce Nesbitt, Bob Pariseau, Rob Peck, Tom Pohorsky, Carl Sassenrath, Carolyn Scheppner, Spencer Shanson, Stan Shepard, Mike Sinz, and Bart Whitebook.

Book production and editing:
Dan Baker

Cover designer:
Hannus Design Associates

C language cross reference:
John Toebes and Doug Walker with SAS C Development System 5.10

Copyright © 1991 by Commodore Electronics Limited.

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book and Addison-Wesley was aware of a trademark claim, the designations have been printed in initial caps. Amiga, Amiga 500, Amiga 1000, Amiga 2000, and Amiga 3000 are registered trademark of Commodore-Amiga, Inc. AmigaDOS, Workbench, and Kickstart are trademarks of Commodore-Amiga, Inc. AUTOCONFIG is a trademark of Commodore Electronics Limited. 68000, 68010, 68020, 68030, 68040, and Motorola are trademarks of Motorola, Inc. ARexx is a trademark of Wishful Thinking Development Corp. Commodore and the Commodore logo are registered trademarks of Commodore Electronics Limited. ADAPTAlphacom is a registered trademark and Alphapro is trademark of Alphacom, Inc. Aztec C and Manx are trademarks of Manx Software Systems. Brother is a registered trademark of Brother Industries, Ltd. Canon is a registered trademark of Canon USA, Inc. CAPE and Inovatronics are trademarks of Inovatronics, Inc. Centronics is a registered trademark of Centronics Data Computer Corp. ColorMaster is a trademark of Cal-Comp. Diablo is a registered trademark of Xerox Corporation. Epson is registered trademark of Epson America, Inc. Hisoft and Devpac Amiga are trademarks of HiSoft. IBM is a registered trademark and Proprinter is a trademark of International Business Machines Corp. Imagewriter and Apple II are trademarks of Apple Computer, Inc. LaserJet and PaintJet are trademarks of the Hewlett Packard Company. Lattice is a registered trademark of Lattice, Inc. LetterPro 20 is a trademark of Qume Corporation. NEC is a registered trademark of NEC Information Systems. Okidata is a registered trademark of Okidata, a division of Oki America, Inc. Okimate 20 is a trademark of Okidata, a division of Oki America, Inc. Pinwriter is a registered trademark of NEC Information Systems. UNIX is a registered trademark of AT&T.

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of the publisher. Printed in the United States of America. Published simultaneously in Canada.

Commodore item number: 327271-07

1 2 3 4 5 6 7 8 9 10 AL 9594939291
First printing, May 1991
ISBN 0-201-56773-3

WARNING: The information described in this manual may contain errors or bugs, and may not function as described. All information is subject to enhancement or upgrade for any reason including to fix bugs, add features, or change performance. As with all software upgrades, full compatibility, although a goal, cannot be guaranteed, and is in fact unlikely.

DISCLAIMER: COMMODORE-AMIGA, INC., ("COMMODORE") MAKES NO WARRANTIES, EITHER EXPRESSED OR IMPLIED, OR REPESENTATIONS WITH RESPECT TO THE INFORMATION DESCRIBED HEREIN. SUCH INFORMATION IS PROVIDED ON AN "AS IS" BASIS AND IS EXPRESSLY SUBJECT TO CHANGE WITHOUT NOTICE. IN NO EVENT WILL COMMODORE BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, OR CONSEQUENTIAL DAMAGES RESULTING FROM ANY CLAIM ARISING OUT OF THE INFORMATION PRESENTED HEREIN, EVEN IF IT HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. SOME STATES DO NOT ALLOW THE EXCLUSION OR LIMITATION OF IMPLIED WARRANTIES OR LIABILITIES FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES, SO THE ABOVE LIMITATION OR EXCLUSION MAY NOT APPLY.

CONTENTS

Introduction 1

Library Autodocs 11

asl.library 16
commodities.library 21
cx_lib.library 35
diskfont.library 43
dos.library 47
exec.library 126
expansion.library 182
gadtools.library 193
graphics.library 205
icon.library 273
iffparse.library 280
intuition.library 303
keymap.library 375
layers.library 378
mathieeedoubbas.library 393
mathieeedoubtrans.library 400
mathieeesingbas.library 409
mathieeesingtrans.library 416
mathtrans.library 425
rexxsyslib.library 434
translator.library 440
utility.library 441
workbench.library 455

Device Autodocs 459

audio.device 462
clipboard.device 473
console.device 477
gameport.device 485
input.device 488
keyboard.device 493
narrator.device 497
parallel.device 503
printer.device 509
serial.device 521
timer.device 530
trackdisk.device 536

Resource Autodocs **547**

battclock.resource 550
battmem.resource 552
cia.resource 555
disk.resource 558
FileSystem.resource 562
misc.resource 563
potgo.resource 565

Linker Library Autodocs **567**

amiga.library 570
debug.library 581
amiga.library source code 585

C Language Include Files (.h) **589**

Clib 590
Devices 616
Dos 638
Graphics 667
Hardware 685
Libraries 721
Resources 741
Rexx 748
Utility 753
Workbench 755

Assembly Language Include Files (.i) **759**

Devices 760
Dos 783
Exec 797
Graphics 817
Hardware 833
Intuition 839
Libraries 866
Resources 883
Rexx 890
Utility 896
Workbench 898

Reference Charts **901**

Function Offsets 902
Structure Reference 913
C Language Cross Reference 924
Amiga Function Index 996

Introduction

About this book

The Amiga Technical Reference Series is the official guide to programming Commodore's Amiga computers. This volume, the *Amiga ROM Kernel Reference Manual: Includes and Autodocs,* contains alphabetically organized Autodoc function summaries and listings of the Amiga system include files.

Autodocs are descriptions of the Amiga's system functions. The include files list the Amiga's system definitions and data structures. This revised edition has been updated for Release 2 of the Amiga operating system.

This manual is the essential quick reference for all Amiga programmers. It contains:

- ☐ Summaries of the system library functions
- ☐ Summaries of the system device commands
- ☐ Summaries of system resource calls
- ☐ Summaries of the amiga.lib functions and source code
- ☐ C language include files (.h files)
- ☐ Assembly language include files (.i files)
- ☐ Handy charts designed to ease debugging and exploring

The other manuals in this series are the *Amiga User Interface Style Guide,* the *Amiga ROM Kernel Reference Manual: Libraries,* the *Amiga ROM Kernel Reference Manual: Devices,* and the *Amiga Hardware Reference Manual.*

About the examples

Except as noted, 68000 assembly language examples have been assembled under the Innovatronics CAPE assembler V2.x, the HiSoft Devpac assembler V1.2, and the Lake Forest Logic ADAPT assembler 1.0. No substantial changes should be required to switch between assemblers.

C examples have been compiled under SAS C, version 5.10a and Manx Aztec C 68K, version 5.0d. Default compiler options are used in both cases. All the C examples assume that the automatic Ctrl-C feature of the compiler has been disabled. Add this code to each example to complete it:

For SAS C (and Lattice C revisions 4.0 and greater)

Compile with: `lc -L <filename>.c`

```
/* Add this function before main(). This overrides the default
* Lattice CTRL-C trap. If this function  returns zero, then the
* CTRL-C event will be ignored */

int CXBRK ( void )    { return(0); }
int chkabort( void ) { return(0); }
```

For Aztec C

Compile with:
```
cc <filename>.c
ln <filename.o> -lc
```

```
/* Add this near the top */

      #include <functions.h>

/* Add this before main() */
      extern int Enable_Abort;    /* reference abort enable */

/* Add this after main(), as the first active line in the program */
      Enable_Abort=0;          /* turn off CTRL-C */
```

General Amiga development guidelines

The environment of the Amiga computer is quite different than that of many other systems. The Amiga is a multitasking platform, which means multiple programs can run on a single machine simultaneously. However, for multitasking to work correctly, care must be taken to ensure that programs do not interfere with one another. It also means that certain guidelines must be followed during programming.

☐ Check for memory loss. Operate your program, then exit. Write down the amount of free memory. Repeat the operation of your program and exit. The amount of free memory remaining should be exactly the same. Any difference indicates a serious problem in your cleanup. (Beware when checking the amount of free memory - some memory loss is normal the first time you open a device or disk-based library because the system has to allocate memory to accommodate them. That is why you should run the program once before checking free memory.)

☐ Use all of the program debugging and stress tools that are available when writing and testing your code. New debugging tools such as Enforcer, MungWall, and Scratch can help find uninitialized pointers, attempted use of freed memory and misuse of scratch registers or condition codes (even in programs that appear to work perfectly).

☐ Always make sure you actually get any system resource that you ask for. This applies to memory, windows, screens, file handles, libraries, devices, ports, etc. Where an error value or return is possible, ensure that there is a reasonable failure path. Many poorly written programs will appear to be reliable, until some error condition (such as memory full or a disk problem) causes the program to continue with an invalid or null pointer, or branch to untested error handling code.

☐ Always clean up after yourself. This applies for both normal program exit and program termination due to error conditions. Anything that was opened must be closed, anything allocated must be deallocated. It is generally correct to do closes and deallocations in reverse order of the opens and allocations. Be sure to check your development language manual and startup code; some items may be closed or deallocated automatically for you, especially in abort conditions. If you write in the C language, make sure your code handles Ctrl-C properly.

☐ Remember that memory, peripheral configurations, and ROMs differ between models and between individual systems. Do not make assumptions about memory address ranges, storage device names, or the locations of system structures or code. Never call ROM routines directly. Beware of any example code you find that calls routines at addresses in the $F00000 - $FFFFFF range. These are ROM routines and they will move with every OS release. The only supported interface to system ROM code is through the library, device, and resource calls.

☐ Never assume library bases or structures will exist at any particular memory location. The only absolute address in the system is $00000004, which contains a pointer to the exec.library base. Do not modify or depend on the format of private system structures. This includes the poking of copper lists, memory lists, and library bases.

☐ Never assume that programs can access hardware resources directly. Most hardware is controlled by system software that will not respond well to interference from other programs. Shared hardware requires programs to use the proper sharing protocols. Use the defined interface; it is the best way to ensure that your software will continue to operate on future models of the Amiga.

☐ Never access shared data structures directly without the proper mutual exclusion (locking). Remember that other tasks may be accessing the same structures.

☐ The system does not monitor the size of a program's stack. Take care that your program does not cause stack overflow, and provide enough extra stack space for the possibility that future revisions of system functions might require additional stack space.

☐ Never use a polling loop to test signal bits. If your program waits for external events like menu selection or keystrokes, do not bog down the multitasking system by busy-waiting in a loop. Instead, let your task go to sleep by Wait()ing on its signal bits. For example:

```
signals = (ULONG)Wait(  (1<<windowPtr->UserPort->mp_SigBit) |
                        (1<<consoleMsgPortPtr->mp_SigBit)     );
```

This turns the signal bit number for each port into a mask, then combines them as the argument for the exec.library/Wait() function. When your task wakes up, handle all of the messages at each port where the SigBit is set. There may be more than one message per port, or no messages at the port. Make sure that you ReplyMsg() to all messages that are not replies themselves. If you have no signal bits to Wait() on, use Delay() or WaitTOF() to provide a measured delay.

☐ Tasks (and processes) execute in 680x0 user mode. Supervisor mode is reserved for interrupts, traps, and task dispatching. Take extreme care if your code executes in supervisor mode. Exceptions while in supervisor mode are deadly.

☐ Most system functions require a particular execution environment. All DOS functions and any functions that might call DOS (such as the opening of a disk-resident library, font, or device) can only be executed from a process. A task is not sufficient. Most other ROM kernel functions may be executed from tasks. Only a few may be executed from interrupts.

☐ Never disable interrupts or multitasking for long periods. If you use Forbid() or Disable(), you should be aware that execution of any system function that performs the Wait() function will temporarily suspend the Forbid() or Disable() state, and allow multitasking and interrupts to occur. Such functions include almost all forms of DOS and device I/O, including common "stdio" functions like printf().

☐ Never tie up system resources unless it is absolutely necessary. For example, if your program does not require constant use of the printer, open the printer.device only when you need it. This will allow other tasks to use the printer while your program is running.You must provide a reasonable error response if a resource is not available when you need it.

☐ All data for the custom chips must reside in Chip memory (type `MEMF_CHIP`). This includes bitplanes, sound samples, trackdisk buffers, and images for sprites, bobs, pointers, and gadgets. The AllocMem() call takes a flag for specifying the type of memory. A program that specifies the wrong type of memory may appear to run correctly because many Amigas have only Chip memory. (On all models of the Amiga, the first 512K of memory is Chip memory and in some later models, Chip memory may occupy the first one or two megabytes).

However, once expansion memory has been added to an Amiga (type `MEMF_FAST`), any memory allocations will be made in the expansion memory area by default. Hence, a program can run correctly on an unexpanded Amiga which has only Chip memory while crashing on an Amiga which has expanded memory. A developer with only Chip memory may fail to notice that memory was incorrectly specified.

Most compilers have options to mark specific data structures or object modules so that they will load into Chip RAM. Some older compilers provide the Atom utility for marking object modules.

If this method is unacceptable, use the AllocMem() call to dynamically allocate Chip memory, and copy your data there.

When making allocations that do not require Chip memory, do not explicitly ask for Fast memory. Instead ask for memory type `MEMF_PUBLIC` or `0L` as appropriate. If Fast memory is available, you will get it.

❑ Never use software delay loops! Under the multitasking operating system, the time spent in a loop can be better used by other tasks. Even ignoring the effect it has on multitasking, timing loops are inaccurate and will wait different amounts of time depending on the specific model of Amiga computer. The timer.device provides precision timing for use under the multitasking system and it works the same on all models of the Amiga. The AmigaDOS Delay() function or the graphics.library/WaitTOF() function provide a simple interface for longer delays. The 8520 I/O chips provide timers for developers who are bypassing the operating system (see the *Amiga Hardware Reference Manual* for more information).

❑ Always obey structure conventions!

All non-byte fields must be word-aligned. Longwords should be longword-aligned for performance.

All address pointers should be 32 bits (not 24 bits). The upper byte must never be used for data.

Fields that are not defined to contain particular initial values must be initialized to zero. This includes pointer fields.

All reserved or unused fields must be initialized to zero for future compatibility.

Data structures to be accessed by the custom chips, public data structures (such as a task control block), and structures which must be longword aligned must *not* be allocated on a program's stack.

Dynamic allocation of structures with AllocMem() provides longword aligned memory of a specified type with optional initialization to zero, which is useful in the allocation of structures.

For 68010/68020/68030/68040 compatibility

Special care must be taken to be compatible with the entire family of 68000 processors:

☐ Do not use the upper 8 bits of a pointer for storing unrelated information. The 68020, 68030, and 68040 use all 32 bits for addressing.

☐ Do not use signed variables or signed math for addresses.

☐ Do not use software delay loops, and do not make assumptions about the order in which asynchronous tasks will finish.

☐ The stack frame used for exceptions is different on each member of the 68000 family. The type identification in the frame must be checked! In addition, the interrupt autovectors may reside in a different location on processors with a VBR register.

☐ Do not use the `MOVE SR,<dest>` instruction! This 68000 instruction acts differently on other members of the 68000 family. If you want to get a copy of the processor condition codes, use the exec.library/GetCC() function.

☐ Do not use the `CLR` instruction on a hardware register which is triggered by Write access. The 68020 `CLR` instruction does a single Write access. The 68000 `CLR` instruction does a Read access first, then a Write access. This can cause a hardware register to be triggered twice. Use `MOVE.x #0, <address>` instead.

☐ Self-modifying code is strongly discouraged. All 68000 family processors have a pre-fetch feature. This means the CPU loads instructions ahead of the current program counter. Hence, if your code modifies or decrypts itself just ahead of the program counter, the pre-fetched instructions may not match the modified instructions. The more advanced processors prefetch more words. If self-modifying code must be used, flushing the cache is the safest way to prevent troubles.

☐ The 68020, 68030 and 68040 processors all have instruction caches. These caches store recently used instructions, but do not monitor writes. After modifying or directly loading instructions, the cache must be flushed. See the exec.library/CacheClearU() Autodoc for more details.

If your code takes over the machine, flushing the cache will be trickier. You can account for the current processors, and hope the same techniques will work in the future:

```
CACRF_ClearI      EQU   $0008       ;Bit for clear instruction cache
                  ;Supervisor mode only.  Use only if you have taken
                  ;over the machine.  Read and store the ExecBase
                  ;processor AttnFlags flags at boot time, call this
                  ;code only if the "68020 or better" bit is set.
ClearICache:      dc.w  $4E7A,$0002       ;MOVEC CACR,D0
                  tst.w d0                ;movec does not affect CC's
                  bmi.s cic_040           ;A 68040 with enabled cache!
                  ori.w #CACRF_ClearI,d0
                  dc.w  $4E7B,$0002       ;MOVEC D0,CACR
                  bra.s cic_exit
cic_040:          dc.w  $f4b8             ;CPUSHA (IC)
cic_exit:
```

Hardware programming guidelines

If you find it necessary to program the hardware directly, then it is your responsibility to write code that will work correctly on the various models and configurations of the Amiga. Be sure to properly request and gain control of the hardware resources you are manipulating, and be especially careful in the following areas:

☐ Kickstart 2.0 uses the 8520 Complex Interface Adaptor (CIA) chips differently than 1.3 did. To ensure compatibility, you must always ask for CIA access using the cia.resource/AddICRVector() and RemICRVector() functions. Do not make assumptions about what the system might be using the CIA chips for. If you write directly to the CIA chip registers, do not expect system services such as the trackdisk.device to function. If you are leaving the system up, do not read or write to the CIA Interrupt Control Registers directly; use the cia.resource/AbleICR(), and SetICR() functions. Even if you are taking over the machine, do not assume the initial contents of any of the CIA registers or the state of any enabled interrupts.

☐ All custom chip registers are Read-only or Write-only. Do not read Write-only registers, and do not write to Read-only registers.

☐ Never write data to, or interpret data from the unused bits or addresses in the custom chip space. To be software-compatible with future chip revisions, all undefined bits must be set to zeros on writes, and must be masked out on reads before interpreting the contents of the register.

☐ Never write past the current end of custom chip space. Custom chips may be extended or enhanced to provide additional registers, or to use bits that are currently undefined in existing registers.

☐ Never read, write, or use any currently undefined address ranges or registers. The current and future usage of such areas is reserved by Commodore and is subject to change.

☐ Never assume that a hardware register will be initialized to any particular value. Different versions of the OS may leave registers set to different values. Check the *Amiga Hardware Reference Manual* to ensure that you are setting up all the registers that affect your code.

Additional assembler development guidelines

If you are writing in assembly language there are some extra rules to keep in mind in addition to those listed above.

☐ Never use the `TAS` instruction on the Amiga. System DMA can conflict with this instruction's special indivisible read-modify-write cycle.

☐ System functions must be called with register A6 containing the library or device base. Libraries and devices assume A6 is valid at the time of any function call. Even if a particular function does not currently require its base register, you must provide it for compatibility with future system software releases.

☐ Except as noted, system library functions use registers D0, D1, A0, and A1 as scratch registers and you must consider their former contents to be lost after a system library call. The contents of all other registers will be preserved. System functions that provide a result will return the result in D0.

☐ Never depend on processor condition codes after a system call. The caller must test the returned value before acting on a condition code. This is usually done with a `TST` or `MOVE` instruction.

Commodore Applications and Technical Support (CATS)

Commodore maintains a technical support group dedicated to helping developers achieve their goals with the Amiga. Currently, technical support programs are available to meet the needs of both smaller, independent software developers and larger corporations. Subscriptions to Commodore's technical support publication, Amiga Mail, is available to anyone with an interest in the latest news, Commodore software and hardware changes, and tips for developers.

To request an application for Commodore's developer support program, or a list of CATS technical publications send a self-addressed, stamped, 9" x 12" envelope to:

CATS-Information
1200 Wilson Drive
West Chester, PA 19380-4231

Error Reports

In a complex technical manual, errors are often found after publication. When errors in this manual are found, they will be corrected in a subsequent printing. Updates will be published in *Amiga Mail*, Commodore's technical support publication.

Bug reports can be sent to Commodore electronically or by mail. Submitted reports must be clear, complete, and concise. Reports must include a telephone number and enough information so that the bug can be quickly verified from your report (i.e., please describe the bug and the steps that produced it).

Amiga Software Engineering Group
ATTN: BUG REPORTS
Commodore Business Machines
1200 Wilson Drive
West Chester, PA 19380-4231
USA

BIX: amiga.com/bug.reports	(Commercial developers)
amiga.cert/bug.reports	(Certified developers)
amiga.dev/bugs	(Others)

USENET: bugs@commodore.COM or uunet!cbmvax!bugs

Library Autodocs

Most of the Amiga's operating system is divided into groups of functions called libraries. Libraries reside either in the Amiga's ROM or on disk. Once a library is opened, any of its functions can be called. For example, if you want to call the Read() function to read data from disk you must first open the DOS library. In general, to do anything useful on the Amiga you must first open a library.

This section contains summaries for all the functions in all of the libraries that are built into the Amiga's operating system software. These summaries have been automatically extracted from the original source code and are called Autodocs (for automatic documentation).

The Autodoc files are organized alphabetically by library, one document for each function call. For additional tutorial information on each of the libraries and more details about the Amiga's library mechanism refer to the *Amiga ROM Kernel Reference Manual: Libraries*. Only a brief introduction is given here. The exec.library is the system's master library and is always open. One Exec function, OpenLibrary(), is used to open all other libraries.

Usage is as follows:

```
struct LibBase *LibBase;      /* Global: declare this above main() */

LibBase = OpenLibrary("library.name",version);
if(!LibBase){ /* Library did not open, so exit */ }
else        { /* Library opened, use functions */ }
```

where:

`LibBase` is a pointer to the library base. This is a global variable used by the system to handle function calls. In the code above, if the library cannot open for some reason LibBase will equal 0. The variable name for a library base pointer is different for each library. Refer to the list below for the appropriate name.

`library.name` is a string that describes the name of the library you wish to open. The list of library names is given below.

`version` should be set to the earliest acceptable library version. A value of 0 matches any version. A value of 33 means you require at least version 33 or a later version of the library. If you specify a higher version number than is present in the system, OpenLibrary() will fail (return 0). For the system libraries, the following table applies:

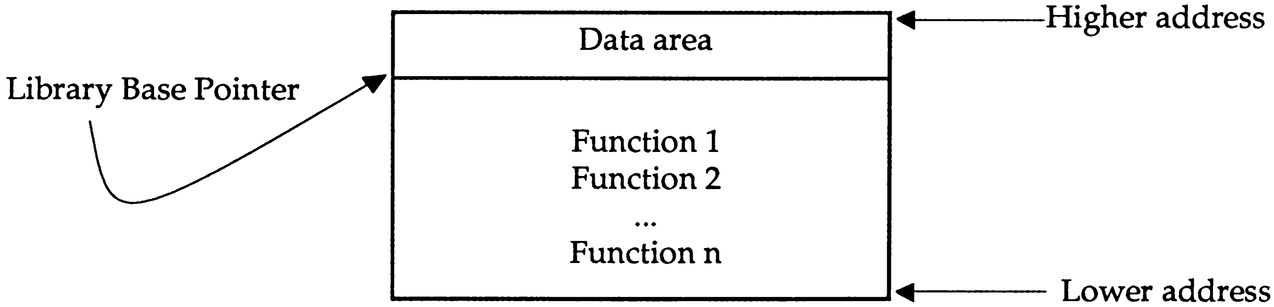
System library version number	Kickstart release
0	Any version
30	Kickstart V1.0 (obsolete)
31	Kickstart V1.1 (NTSC only - obsolete)
32	Kickstart V1.1 (PAL only - obsolete)
33	Kickstart V1.2 (the oldest revision still in use)
34	Kickstart V1.3 (adds autoboot to V33)
35	Special Kickstart version to support A2024 high-resolution monitor
36	Kickstart V2.00
37	Kickstart V2.04

Many of the libraries and functions documented in this manual are available in all versions of the Amiga operating system. Others are completely new and cannot be used unless you have successfully opened the appropriate version of the library.

The functions which are new for V36 or V37 are marked with `(V36)` or `(V37)` in the NAME line of the function Autodoc. These new functions are not available in earlier versions of the Amiga libraries. If your code calls any system function that is marked with `(V36)` or `(V37)` then use a matching version number (36, 37, or higher) when opening the library.

Exit gracefully and informatively if the required library version is not available. Please note that since V34 Kickstart is nearly identical to V33, it is generally *not* wise to require V34.

The OpenLibrary() function returns the address of the library base, which you must assign to a specific global system variable. The list of global system variable names for each library is given below (case is important.) The system uses the library base to access the functions of the library. If the library cannot open for some reason, the OpenLibrary() function returns zero. Library bases represent a midpoint in the library. Below the base are the function vectors, above the base is the library data area:



The names of the libraries that are currently part of the Amiga system software and their associated library base pointer names are:

Library Name	**Library Base Pointer Name**	
asl.library	AslBase	V37
commodities.library	CxBase	V37
diskfont.library	DiskfontBase	
dos.library	DOSBase	
exec.library	SysBase	
expansion.library	ExpansionBase	
gadtools.library	GadToolsBase	V37
graphics.library	GfxBase	
icon.library	IconBase	
iffparse.library	IFFParseBase	V37
intuition.library	IntuitionBase	
keymap.library	KeymapBase	
layers.library	LayersBase	
mathffp.library	MathBase	
mathtrans.library	MathTransBase	
mathieeedoubbas.library	MathIeeeDoubBasBase	
mathieeedoubtrans.library	MathIeeeDoubTransBase	
mathieeesingbas.library	MathIeeeSingBasBase	
mathieeesingtrans.library	MathIeeeSingTransBase	
rexxsyslib.library	RexxSysBase	V37
translator.library	TranslatorBase	
utility.library	UtilityBase	V37
workbench.library	WorkbenchBase	

All Amiga libraries accept parameters in registers, and return the result in data register D0. All routines return a full 32 bit longword unless noted otherwise in the function Autodoc. This allows programs and functions that are written in assembler to communicate quickly. It also eliminates the dependence on the stack frame conventions of any particular language. Some C language compilers for the Amiga can generate parameters directly into registers, others translate any Amiga library call into a stub routine that moves parameters from the stack to registers. See the discussion of "amiga.lib" in the Linker Libraries section of this book for more details.

Complete examples of how to open and close a library follow.

```
/*
* A complete ready-to-compile example of library use.
* The library is opened, checked, used and closed.
* See the intuition.library document for a description
* of what the DisplayBeep() function does.
*
*/

struct Library *OpenLibrary();            /* declare return type */

struct IntuitionBase *IntuitionBase;      /* get storage for base */

void main()
{
  IntuitionBase=(struct IntuitionBase *)
      OpenLibrary("intuition.library",33L);

  if(!IntuitionBase)          /* check if it actually opened */
      exit(20);               /* a return value of 20 indicates failure */

      DisplayBeep(0L);        /* use the library function */

      CloseLibrary(IntuitionBase);
}
```

```
**********************************************************************
*
* A complete ready-to-assemble example of library use.   The intuition
* library is opened, checked, used, and closed.  See the intuition
* document for a description of what the DisplayBeep() function does.
*
* When calling an Amiga library, the base pointer *must* be in
* A6... the library is free to depend on this.  Registers D0,D1,A0
* and A1 may be destroyed by the library, all others will be preserved.
*

        _AbsExecBase      EQU 4     ;Where exec's library base is
XREF    _LVOOpenLibrary             ;Offset from base for OpenLibrary
XREF    _LVOCloseLibrary            ;  "
XREF    _LVODisplayBeep             ;  "

        move.l  _AbsExecBase,a6       ;Move exec.library base to a6
        lea.l   IntuiName(pc),a1      ;Pointer to "intuition.library"
        moveq   #33,d0                ;Version
        jsr     _LVOOpenLibrary(a6)   ;Call exec's OpenLibrary()
        tst.l   d0
        bne.s   open_ok
        moveq   #20,d0                ;Set failure code
        rts                           ;Failed exit

open_ok move.l  d0,a6                 ;Put IntuitionBase in a6.
        suba.l  a0,a0                 ;Load zero into a0
        jsr     _LVODisplayBeep(a6)   ;Call intuition's DisplayBeep()

        move.l  a6,a1                 ;Put IntuitionBase into a1
        move.l  _AbsExecBase,a6
        jsr     _LVOCloseLibrary(a6)  ;Call exec's CloseLibrary()
moveq   #0,d0                         ;Set return code
        rts

IntuiName:  dc.b 'intuition.library',0
        END
```

TABLE OF CONTENTS

asl.library/AllocAslRequest
asl.library/AllocFileRequest
asl.library/AslRequest
asl.library/FreeAslRequest
asl.library/FreeFileRequest
asl.library/RequestFile

```
asl.library/AllocAslRequest                          asl.library/AllocAslRequest

   NAME
        AllocAslRequest -- alloc an ASL requester, with TagItem modifiers (V36)

   SYNOPSIS
        request = AllocAslRequest( type, ptags )
        D0                         D0    A0

        APTR    request;
        ulong   type;
        struct TagItem  *ptags;

   FUNCTION
        Allocates an ASL requester data structure of the specified type,
        with optional TagItem modifiers.

   INPUTS
        type = type of requester to create.  Currently defined types
            include ASL_FileRequest and ASL_FontRequest.
        ptags = pointer to a tagitem array, which is defined for each
            specified type.  See "asl.h" and example programs for usage
            of various tag types.  See AslRequest() for specifications
            of currently defined tag values and their effects.

        Ncte that tag values stay in effect for each use of the
        requester until they are cleared or modified by passing the
        same tag with a new value.

        AllocAslRequestTags( type, tags... ) which accepts your tags
        on the stack, is available in amiga.lib.

        Example Usage: AllocAslRequestTags( ASL_FileRequest,
                                            ASL_Hail, "My Title Bar",
                                            TAG_DONE );

   RESULT
        Pointer to an initialized requester data structure, or NULL on
        failure.  The data structure returned will match the requested
        type; for type ASL_FileRequest, a struct FileRequester *; for
        ASL_FontRequest, a struct FontRequester *.

        The requester returned may then be passed to AslRequest(), and
        is freed by calling FreeAslRequest().

   SEE ALSO
        AslRequest(), FreeAslRequest()
```

```
asl.library/AllocFileRequest                        asl.library/AllocFileRequest

   NAME
        AllocFileRequest -- allocates a FileRequester structure (V36)

   SYNOPSIS
        request = AllocFileRequest()
        D0

        struct FileRequester *request;

   FUNCTION
        Creates and initializes the data structure required to pass to the
        RequestFile() function.

   INPUTS
        None.  If you wish to get other than default values, you can use
        AllocAslRequest() to set up a file request with tag items.

   RESULT
        Pointer to a struct FileRequester, which is to be passed to the
        RequestFile() function.

        The returned FileRequester pointer has public fields which are
        readable by the application as defined in aslbase.h.

   CAUTION
        The application MUST use either the AllocFileRequest(), or
        AllocAslRequest(), function to allocate the structure to be passed to
        the FileRequest() or AslRequest() functions; it is not possible to
        create a struct FileRequest except through the library calls.

        Also, any modifications MUST be done through TagItem values, rather
        than directly modifying, unless explicitly documented otherwise.

   SEE ALSO
        RequestFile(), FreeAslRequest(), AslRequest()
```

```
asl.library/AslRequest                                    asl.library/AslRequest

   NAME
        AslRequest -- get input from user for an ASL requester (V36)

   SYNOPSIS
        BOOL result = AslRequest( request, ptags );
        D0                         A0       A1

        BOOL    result;
        APTR    request;
        struct TagItem  *ptags;

   FUNCTION
        Prompts the user for input, based on the specific type of
        requester and modifying tagitems.  The actions and results
        are specific to the type but in general the action is to open
        a requesting window prompting the user for a specific input.
        If the user cancels or the system aborts the request, NULL
        is returned, otherwise the request data structure readable
        data reflects the user input.
        Note that tag values stay in effect for each use of the
        requester until they are cleared or modified by passing the
        same tag with a new value.

   INPUTS
        request = requester structure allocated with AllocAslRequest().
        ptags = pointer to an array of TagItems which may be used to
            modify the requester.

        AslRequestTags( type, tags... ) which accepts your tags
        on the stack, is available in amiga.lib.

        Example Usage: AslRequestTags( ASL_FileRequest,
                                       ASL_Hail, "My Title Bar",
                                       TAG_DONE );

   TAGS

        ( REMEMBER - ALL DATA STRUCTURES ARE READ-ONLY EXCEPT BY USING
          TAGITEMS !!! )

        ASL_Hail (STRPTR) - Hailing text to prompt user, typically
            displayed in window title bar.
        ASL_Window (struct Window *) - Parent window for the request
            function, which is used to select the screen on which the
            requesting window will be displayed and also is used for
            a shared IDCMP port.

        ASL_LeftEdge (WORD) - Preferred display position for left edge where
            request window should open.
        ASL_TopEdge (WORD) - Preferred top edge of request window.
        ASL_Width (WORD)   - Preferred width of request window.
        ASL_Height (WORD)  - Preferred height of request window.

        ASL_HookFunc (APTR) - Pointer to callback function, specific to
            each AslRequest type.

        ASL_File (STRPTR) - FileRequester initial filename string.
        ASL_Dir (STRPTR) - FileRequester initial directory path string.

        ASL_FontName (STRPTR) - FontRequester initial fontname string.
        ASL_FontHeight (UWORD) - FontRequester initial height (ta_YSize).
        ASL_FontStyles (UBYTE) - FontRequester initial styles (ta_Style).
        ASL_FontFlags (UBYTE) - FontRequester initial flags (ta_Flags).
```

```
ASL_FrontPen (BYTE) - FontRequester front pen color (fo_FrontPen).
ASL_BackPen (BYTE) - FontRequester back pen color (fo_BackPen).
ASL_MinHeight (UWORD) - Minimum height for FontRequester display
    of font sizes.  (Application must check return value).
ASL_MaxHeight (UWORD) - Maximum height for FontRequester display
    of font sizes. (Application must check ta_YSize returned).

ASL_OKText (STRPTR) - Replacement for default "OK" gadget text.
    ( Limited to approx. six characters ).
ASL_CancelText (STRPTR) - Replacement for default "CANCEL"
    gadget text. ( Limited to approx. six characters ).

ASL_FuncFlags (ULONG) - Function flags, depends on requester
    type.  Example: FILF_SAVE for FileRequester.
ASL_ExtFlags1 (ULONG) - Extended flags (to pass FILF1_ bitdefs)
           Example: FIL1F_NOFILES for file requester
```

RESULT

If NULL, typically the user cancelled the requester or a system failure prevented the requester from being opened. If non-zero, values will be set depending on the particular type of request, in the requesting data structure. See "libraries/asl.h" for information on the READ-ONLY fields in each specific type of requester.

NOTES

Asl provides a way for applications to interact with requester operation via a callback (hook) function. For the ASL file and font requesters, there are two ASL_FuncFlags to specify that you want a callback:

```
for FileRequester: FILF_DOWILDFUNC and FILF_DOMSGFUNC
for FontRequester: FONF_DOWILDFUNC and FONF_DOMSGFUNC
```

The DOWILDFUNC allows you to perform the pattern matching. The DOMSGFUNC allows you to handle IDCMP messages received for windows that are sharing a UserPort with the requester.

If you set one or both of these flags via the ASL_FuncFlags tagitem, you must provide a pointer to your hook function using the ASL_HookFunc tagitem. Your function will be called as follows:

```
 ULONG rf_Function(ULONG Mask, CPTR Object, CPTR AslRequester)
```

The Mask value is a copy of the specific ASL_FuncFlag value the callback is for. Object is a pointer to a data object specific to the reason for the callback (defined by Mask). AslRequester is a pointer to the requester structure.

Note that you can only define one HookFunc per requester. Your hook function must examine the Mask passed to it to determine what the callback is for and what the Object is.

The following table will explain what is passed to, and expected to be returned by a hook functions for various masks:

```
FileRequester DOWILDFUNC
Purpose: to accept or reject individual files for display list
Inputs: Mask            = FILF_DOWILDFUNC
        Object          = struct AnchorPath *
        AslRequester    = struct FileRequester *
Result: You return zero to accept file for display in list
```

```
FontRequester DOWILDFUNC
Purpose: to accept or reject individual fonts for display list
Inputs: Mask            = FONF_DOWILDFUNC
        Object          = struct TextAttr *
        AslRequester    = struct FontRequester *
Result: You return non-zero to accept font for display in list


FileRequester (or FontRequester) DOMSGFUNC
Purpose: to handle IDCMP msgs for other windows sharing port
Inputs: Mask            = FILF_DOMSGFUNC (FONF_DOMSGFUNC)
        Object          = struct IntuiMessage *
        AslRequester    = struct FileRequester (FontRequester) *
Result: You must return the Object pointer (asl will Reply the Object)
```

SEE ALSO

AllocAslRequest(), FreeAslRequest()

```
asl.library/FreeAslRequest                              asl.library/FreeAslRequest

   NAME
        FreeAslRequest - frees requester obtained from AllocAslRequest (V36)

   SYNOPSIS
        FreeAslRequest( request )
                        A0

        APTR request;

   FUNCTION
        FreeAslRequest() is used to free the structure returned by
        AllocAslRequest() or AllocFileRequest(), in order to free
        all resources associated with that requester after the
        application has completed all use of the data structures.

   INPUTS
        request - value returned from AllocAslRequest() or
                        AllocFileRequest().

   RESULT
        None.  All resources associated with the request will be
        freed.

   SEE ALSO
        AllocAslRequest(), AslRequest(), AllocFileRequest()
```

```
asl.library/FreeFileRequest                            asl.library/FreeFileRequest

   NAME
        FreeFileRequest -- frees requester allocated by AllocFileRequest (V36)

   SYNOPSIS
        FreeFileRequest( request )
                         A0

        struct FileRequester *request;

   FUNCTION
        This function is identical to the FreeAslRequest() function, but is
        documented for source code compatability and ease of use.
        Applications may use either FreeAslRequest() or FreeFileRequest() to
        free the data structures allocated by AllocFileRequest().

   INPUTS
        request = the return value from AllocFileRequest().

   SEE ALSO
        FreeAslRequest()
```

```
asl.library/RequestFile                              asl.library/RequestFile

   NAME
        RequestFile -- request user to select file(s) (V36)

   SYNOPSIS
        BOOL result = RequestFile( request )
        D0                         A0

        BOOL    result;
        struct FileRequester *request;

   FUNCTION
        RequestFile() displays a file requester and waits for the user to
        select filenames or cancel the request.  This function is identical
        to the AslRequest() function, except that there is no TagList to
        modify the settings for the requester.  See AslRequest() for details.

   INPUT
        request = struct FileRequester * returned by AllocFileRequest().

   RESULT
        result - See AslRequest() result.  NULL indicates cancelled.

   SEE ALSO
        AllocFileRequest(), FreeFileRequest(), AslRequest()
```

TABLE OF CONTENTS

commodities.library/ActivateCxObj
commodities.library/AddIEvents
commodities.library/AttachCxObj
commodities.library/ClearCxObjError
commodities.library/CreateCxObj
commodities.library/CxBroker
commodities.library/CxMsgData
commodities.library/CxMsgID
commodities.library/CxMsgType
commodities.library/CxObjError
commodities.library/CxObjType
commodities.library/DeleteCxObj
commodities.library/DeleteCxObjAll
commodities.library/DisposeCxMsg
commodities.library/DivertCxMsg
commodities.library/EnqueueCxObj
commodities.library/InsertCxObj
commodities.library/InvertKeyMap
commodities.library/ParseIX
commodities.library/RemoveCxObj
commodities.library/RouteCxMsg
commodities.library/SetCxObjPri
commodities.library/SetFilter
commodities.library/SetFilterIX
commodities.library/SetTranslate

```
commodities.library/ActivateCxObj           commodities.library/ActivateCxObj

   NAME
     ActivateCxObj() (V36)

   SYNOPSIS
     previous = ActivateCxObj(co, true);

     LONG ActivateCxObj(CxObj *,LONG);

   FUNCTION
     Commodities objects of all types maintain an "activation state."
     If an object is "active," then it performs its particular action
     whenever a Commodities message arrives.  If the object is
     "inactive," no action is taken, and the message goes on to its
     next destination.

     All objects are created in the "active" state except brokers,
     which are created "inactive."  Thus, after you create your
     broker and hang a tree of objects off of it, you must remember
     to use this function to activate your broker.  This causes it
     it to divert all messages to your tree of objects.

     This function activates 'co' if 'true,' and deactivates it
     otherwise.  The previous value of activation is returned:
     zero for inactive, non-zero for active.

   DIAGNOSTICS
     None.

   SEE ALSO
     CxBroker()
     Brokers and Application Sub-Trees (in Reference Manual)
```

commodities.library Page 3

```
commodities.library/AddIEvents                 commodities.library/AddIEvents

  NAME
    AddIEvents()  (V36)

  SYNOPSIS
    AddIEvents(ie)

    void AddIEvents(struct InputEvent *);

  FUNCTION
    This function adds a null-terminated linked list to the input
    stream of Commodities.  It is a touch easier than using the
    input device directly.

    The contents of the input events are copied into Commodities internal
    messages, so they may be disposed of as soon as this call returns.

    The messages are initially routed to the first Broker in the
    Commodities Object List.

  DIAGNOSTICS
    None.

  SEE ALSO
    FreeIEvents()
```

commodities.library Page 4

```
commodities.library/AttachCxObj               commodities.library/AttachCxObj

  NAME
    AttachCxObj()  (V36)

  SYNOPSIS
    AttachCxObj(headobj, co);

    void AttachCxObj(CxObj *,CxObj *);

  FUNCTION
    Appends object 'co' to the list of object 'headobj,' using the
    Exec function AddTail().  Returns 'headobj.'

  DIAGNOSTICS
    If 'co' is null, this function will record that fact in the
    internal accumulated error of 'headobj.'  This error record
    can be retrieved using CxObjError() and cleared using
    ClearCxObjError().

  SEE ALSO
    exec.library/AddTail()
    Objects and Messages (in Reference Manual)
    CxObjError(), ClearCxObjError()
```

```
commodities.library/ClearCxObjError       commodities.library/ClearCxObjError

   NAME
     ClearCxObjError() (V36)

   SYNOPSIS
    ClearCxObjError(co);

    void ClearCxObjError(CxObj *);

   FUNCTION
     Clears the accumulated error value of Commodities object 'co.'

     It is unwise to do this to a filter if COERR_BADFILTER is set.
     This will fool the Commodities Exchange into thinking the
     filter is OK.  Set another, valid, filter, or leave it alone.

   DIAGNOSTICS
     None.

   SEE ALSO
     CxObjError()
```

```
commodities.library/CreateCxObj                    commodities.library/CreateCxObj

   NAME
     CreateCxObj() (V36)

   SYNOPSIS
     co = CreateCxObj(type, arg1, arg2);

     CxObj *CreateCxObj(ULONG,LONG,LONG);

   FUNCTION
     Creates a Commodities Object of type 'type.'  It is not proper
     to call this function directly.  Each object creation routine
     except CxBroker() is defined as a macro in <libraries/commodities.h>
     The routines are independently documented.

     Note that the handle returned is a pointer to an abstract type
     named 'CxObj.'  This type is defined as a LONG in
     <libraries/commodities.h> but internally has more content.  The
     specific size and contents of internal data structures are 'private.'

     All functions which operate on a Commodities object are made with
     a reference to the thirty-two bit value returned by this function
     (or CxBroker()).

   DIAGNOSTICS
     A NULL returned value indicates that the requested object could not
     be created, typically for lack of system memory.  Minor problems in
     creating an object, such as providing a bad filter description to
     CxFilter(), typically don't cause failure, but are recorded in
     an internal error field in the new object.

   SEE ALSO
     CxObjError(), CxFilter(), CxTypeFilter(), CxSender(), CxSignal(),
     CxTranslate(), CxDebug(), CxCustom(), CxBroker()
```

commodities.library Page 7

```
commodities.library/CxBroker                          commodities.library/CxBroker

   NAME
     CxBroker() (V36)

   SYNOPSIS
     broker = CxBroker(nb, error);

     CxObj *CxBroker(struct NewBroker *,LONG *);

   FUNCTION
     Creates a broker from the specification found in the NewBroker
     structure pointed to by 'nb.'

     The purposes and meaning of the fields are described below.
     See also the include file <libraries/commodities.h>

     struct NewBroker {
        BYTE   nb_Version;
        BYTE   *nb_Name;
        BYTE   *nb_Title;
        BYTE   *nb_Descr;
        SHORT   nb_Unique;
        BYTE   nb_Pri;
     };

     'nb_Version' is the way that future versions of the Commodities
     library can identify which version of the NewBroker structure
     you are using.  This should be set to NB_VERSION, defined in
     <libraries/commodities.h>

     'nb_Name' is a string which is to be the name of the broker
     Node which is created.  This name is used to find the broker Node
     in the Commodities Object List and is the name shown in the list
     gadget in the Exchange program.

     'nb_Title' and 'nb_Descr' are two strings which appear to the
     user and describe the application the broker is representing.
     Note that all strings above are copied into the broker object.
     The maximum length of these strings that will be recognized are
     defined by constants in <libraries/commodities.h>.

     'nb_Unique' is a field that indicates what should happen if
     a broker of the same name (nb_Name) already exists in the
     Commodities Object List. Constants in <libraries/commodities.h>
     allow the caller to specify whether another broker is to be created,
     and whether any existing broker of the same name should be
     notified that an attempt at creating a duplicate has been made.

     'nb_Pri' specifies with what priority in the Commodities
     Object List that the new broker is to be inserted.  Higher
     priority nodes appear earlier in a list. See
     <libraries/commodities.h> for guidelines for priorities of
     different types of applications. It is strongly recommended that
     the ToolTypes environment of an application be used to allow the
     end-user to set the priority of the broker.

   DIAGNOSTICS
     Returns NULL in the event of failure.  If the pointer 'error'
     is non-null, a further diagnostic code will be placed at
     that address.  Error codes are in <libraries/commodities.h> and
     include:

   CBERR_OK
     No problems; broker created OK.
```

commodities.library Page 8

```
   CBERR_SYSERR
     System problems, not your fault.  Sign of low memory or
     big problems.

   CBERR_DUP
     The nb_Unique field specified that only one broker of
     'nb_Name' should be allowed, and one already exists.

   CBERR_VERSION
     The field 'nb_Version' is unknown to the running version
     of "commodities.library."

   SEE ALSO
     Brokers and Application Sub-Trees (in Reference Manual)
```

```
commodities.library/CxMsgData                    commodities.library/CxMsgData

  NAME
    CxMsgData()  (V36)

  SYNOPSIS
    contents = CxMsgData(cxm);

    char *CxMsgData(struct CxMsg *);

  FUNCTION
    Most Commodities messages contain meaningful data, such as an
    InputEvent structure.  A pointer to this data may be obtained by
    using this function.  A valid, non-null pointer will always
    be returned if 'cxm' is valid.

    You may get a Commodities message from a synchronous (custom
    object) or asynchronous (sender object) source.  In the second
    case, 'contents' is not valid after you have replied to the message.

  DIAGNOSTICS
    If 'cxm' is null, returns NULL.

  SEE ALSO
    CxSender()
    CxCustom()
```

```
commodities.library/CxMsgID                          commodities.library/CxMsgID

  NAME
    CxMsgID()  (V36)

  SYNOPSIS
    id = CxMsgID(cxm);

    LONG CxMsgID(struct CxMsg *);

  FUNCTION
    Returns the value associated with the cause or source of the
    Commodities message 'cxm.'  Values are provided by the application
    when a Sender or Custom object is created.

  DIAGNOSTICS
    If not specified by the application, the ID value of a Commodities
    message will be null.  It is suggested that using non-null
    values in your program as a rule may identify some possible errors.

  SEE ALSO
    CxSender()
    CxCustom()
```

commodities.library Page 11

```
commodities.library/CxMsgType                    commodities.library/CxMsgType

  NAME
    CxMsgType() (V36)

  SYNOPSIS
    type = CxMsgType(cxm);

    ULONG CxMsgType(struct CxMsg *);

  FUNCTION
    Returns the type of a Commodities message.  Possible values of
    'type' are found in <libraries/commodities.h>. Most Commodities
    messages are Input Event messages.

  DIAGNOSTICS
    The value NULL should never be returned, if 'cxm' points to
    a valid message.

  SEE ALSO
```

commodities.library Page 12

```
commodities.library/CxObjError                    commodities.library/CxObjError

  NAME
    CxObjError() (V36)

  SYNOPSIS
    error = CxObjError(co);

    LONG CxObjError(CxObj *);

  FUNCTION
    When a function acting on an object fails, it records the failure
    in the object's internal data.  This function returns the
    accumulated error value.  The values are represented by flag
    bits defined in <libraries/commodities.h>. Several errors may be
    recorded by multiple bits in 'error.'

    The possible errors (at current writing) are:

  COERR_ISNULL
    The value of parameter 'co' was in fact NULL.  This error
    means "the problem with the object you inquire about is
    that it failed to be created."

  COERR_NULLATTACH
    Using the Commodities list manipulation functions, an
    attempt was made to add a NULL object to the list belonging
    to 'co.'  This allows a line of code as follows to exist
    in an error-tolerant program:

         AttachCxObj(filter, CxSender(myport, MY_ID));

  COERR_BADFILTER
    The most recent filter specification for a Filter object
    was faulty.  This happens if no sense can be made out
    of a description string, or if an Input Expression (IX)
    has an invalid format or version byte.
    (See <libraries/commodities.h>)

    When this bit is set in a filter's error field, the
    filter will match nothing, but this is not the proper
    way to "turn off" a filter.  Use ActivateCxObj().

  COERR_BADTYPE
    A type specific operation, such as SetFilterIX(), was called
    for object 'co,' but 'co' isn't of the proper type.

  DIAGNOSTICS
    Nothing but.

  SEE ALSO
    SetFilter(), SetFilterIX(), AttachCxObj(), ActivateCxObj(),
    ClearCxObjError()
```

```
commodities.library/CxObjType                    commodities.library/CxObjType

  NAME
    CxObjType() (V36)

  SYNOPSIS
    type = CxObjType(co);

    ULONG CxObjType(CxObj *);

  FUNCTION
    This function should not really be necessary.  It returns
    the type of a Commodities object, which you should already
    know, since you created it in the first place.

    Values for 'type' are given in <libraries/commodities.h>

  DIAGNOSTICS
    The value of 'type' will be CX_INVALID if you pass in a value of
    NULL for 'co.'  If you pass a random value for 'co,' you
    will get a random answer.

  SEE ALSO
    All Commodities object creation routines.
```

```
commodities.library/DeleteCxObj                  commodities.library/DeleteCxObj

  NAME
    DeleteCxObj() (V36)

  SYNOPSIS
    DeleteCxObj(co);

    void DeleteCxObj(CxObj *);

  FUNCTION
    Deletes a Commodities object of any type.  If the object
    is linked into a list, it will first be removed.  Note that
    the handle 'co' is invalid after this function is called.

    Note also that deleting an object which has other objects attached
    to it may be undesirable.  Use the function DeleteCxObjAll()
    to delete an entire sub-tree of objects.

  DIAGNOSTICS
    None.  Possible system crash if fed fiction.

  SEE ALSO
    exec.library/Remove()
    DeleteCxObjAll()
```

commodities.library Page 15

```
commodities.library/DeleteCxObjAll          commodities.library/DeleteCxObjAll

  NAME
    DeleteCxObjAll() (V36)

  SYNOPSIS
    DeleteCxObjAll(co);

    void DeleteCxObjAll(CxObj *);

  FUNCTION
    This function deletes the Commodities object 'co,' and also
    recursively deletes all objects attached to 'co,' and the
    objects attached to them.

    If 'co' is linked into a list, it is removed first.  Note
    that the handle 'co' is invalid after this function is called.

    This function is useful when an application exits: most
    applications can clean up completely by deleting the
    entire sub-tree of objects starting at their broker.

  DIAGNOSTICS
    None.  Definite system crash if fed fiction.

  SEE ALSO
    exec.library/Remove()
    DeleteCxObj()
```

commodities.library Page 16

```
commodities.library/DisposeCxMsg              commodities.library/DisposeCxMsg

  NAME
    DisposeCxMsg() (V36)

  SYNOPSIS
    DisposeCxMsg(cxm);

    void DisposeCxMsg(struct CxMsg *);

  FUNCTION
    Eliminates the Commodities message pointed to by 'cxm.'
    Can be used to 'swallow' InputEvents by disposing of
    every Commodities message of type CXM_IEVENT.

  DIAGNOSTICS
    None.

  SEE ALSO
```

```
commodities.library/DivertCxMsg                    commodities.library/DivertCxMsg

  NAME
    DivertCxMsg() (V36)

  SYNOPSIS
    DivertCxMsg(cxm, headobj, returnobj);

    void DivertCxMsg(struct CxMsg *,CxObj *,CxObj *);

  FUNCTION
    Sends Commodities message 'cxm' down the list of objects attached to
    'headobj.'  The pointer 'returnobj' is first pushed onto the
    routing stack of 'cxm' so that when the end of the list of 'headobj'
    is reached the SUCCESSOR of 'returnobj' is the next destination.

    For example, when a filter finds a match with a message, the
    message is diverted down the filter's list like this:

      DivertCxMsg(cxm, filter, filter);.

  DIAGNOSTICS
    None.

  SEE ALSO
    Reference Manual
```

```
commodities.library/EnqueueCxObj                  commodities.library/EnqueueCxObj

  NAME
    EnqueueCxObj() (V36)

  SYNOPSIS
    EnqueueCxObj(headobj, co);

    void EnqueueCxObj(CxObj *,CxObj *);

  FUNCTION
    Puts object 'co' into the list of object 'headobj,' using the
    Exec function Enqueue().  Returns 'headobj.'

    This function uses the priority of the Commodities object as a node.
    This priority can be set using SetCxObjPri().

  DIAGNOSTICS
    If 'co' is null, will record that fact in the internal accumulated
    error of 'headobj.'  This error record can be retrieved using
    CxObjError() and cleared using ClearCxObjError().

  SEE ALSO
    exec.library/Enqueue()
    SetCxObjPri()
    Objects and Messages (in Reference Manual)
    CxObjError(), ClearCxObjError()
```

commodities.library Page 19

```
commodities.library/InsertCxObj                    commodities.library/InsertCxObj

  NAME
    InsertCxObj() (V36)

  SYNOPSIS
    InsertCxObj(headobj, co, pred);

    void InsertCxObj(CxObj *,CxObj *,CxObj *);

  FUNCTION
    Adds object 'co' to the list of object 'headobj,' using the
    Exec function Insert().  Returns 'headobj.'

    As described in the autodocs for exec.library/Insert(), 'co' will
    be inserted in the list of 'headobj' prior to 'pred.'  Even though
    Insert() doesn't use the list header unless 'pred' is null, calling
    InsertCxObj() you should always pass a valid 'headobj' and a possibly
    null 'pred.'

  DIAGNOSTICS
    If 'co' is null, this will be recorded in the internal accumulated
    error of 'headobj.'  This error record can be retrieved using
    CxObjError() and cleared using ClearCxObjError().

  SEE ALSO
    exec.library/Insert()
    Objects and Messages (in Reference Manual)
    CxObjError(), ClearCxObjError()
```

commodities.library Page 20

```
commodities.library/InvertKeyMap                  commodities.library/InvertKeyMap

  NAME
    InvertKeyMap() (V36)

  SYNOPSIS
    retval = InvertKeyMap(ansicode, ie, km)

    ULONG InvertKeyMap(ULONG,struct InputEvent *,struct KeyMap *);

  FUNCTION
    Uses the system call MapANSI() to figure out what InputEvent
    translates to an ANSI character code 'ansicode.'  The InputEvent
    pointed to by 'ie' is filled in with that information.  The KeyMap
    'km' is used, unless it is NULL, in which case the system standard
    keymap (as defined when commodities.library is initialized) is used.

    This function currently handles one-deep dead keys (such as
    <alt f>o).  It does not look up the high key map (keystrokes
    with scan codes greater than 0x40), and misses system changes to
    the default key map.

  DIAGNOSTICS
    Returns TRUE if it worked, FALSE otherwise.

  SEE ALSO
    InvertString()
```

commodities.library Page 21

```
commodities.library/ParseIX                         commodities.library/ParseIX

   NAME
     ParseIX() (V36)

   SYNOPSIS
     failurecode = ParseIX(string, ix);

     LONG ParseIX(char *,IX *);

   FUNCTION
     Given an input description string and an allocated, valid input
     expression, sets the fields of the input expression to
     correspond to the description string.

   DIAGNOSTICS
     CHANGED FOR v0.3!!!
     Returns zero if the string WAS successfully parsed.

   SEE ALSO
     Input Expressions and Description Strings
```

commodities.library Page 22

```
commodities.library/RemoveCxObj                 commodities.library/RemoveCxObj

   NAME
     RemoveCxObj() (V36)

   SYNOPSIS
     RemoveCxObj(co);

     void RemoveCxObj(CxObj *);

   FUNCTION
     Removes Commodities object 'co' from any list it may be a part of.
     Will not crash if 'co' is null, or if it has not been inserted
     in a list (and is not corrupted).

     It is not recommended to remove a broker from the master list.

   DIAGNOSTICS
     None.

   SEE ALSO
     Objects and Messages (in Reference Manual)
```

commodities.library Page 23

```
commodities.library/RouteCxMsg                      commodities.library/RouteCxMsg

  NAME
    RouteCxMsg() (V36)

  SYNOPSIS
    RouteCxMsg(cxm, co);

    void RouteCxMsg(struct CxMsg *,CxObj *);

  FUNCTION
    Establishes the next destination of a Commodities message
    to be 'co,' which must be a valid Commodities object, and
    must be linked in ultimately to the Commodities Object List.

    Such a routing of an object is analogous to a 'goto' in a
    program.  There is no effect on the message's routing stack.

  DIAGNOSTICS
    None.

  SEE ALSO
    DivertCxMsg()
```

commodities.library Page 24

```
commodities.library/SetCxObjPri                      commodities.library/SetCxObjPri

  NAME
    SetCxObjPri() (V36)

  SYNOPSIS
    SetCxObjPri(co, pri)

    void SetCxObjPri(CxObj *,LONG);

  FUNCTION
    This function sets the priority of a Commodities object for the
    purposes of EnqueueCxObj().  Note that the Commodities list
    mechanisms are based on Amiga Exec lists, so the priority is
    recorded as a signed byte.  Please keep values in range.  A
    value of zero is usually fine.

    It is strongly recommended that the ToolTypes environment be
    utilized to provide end-user control over the priority of
    brokers, but application specific ordering of other objects
    within their lists is not dictated.

  DIAGNOSTICS
    None.

  SEE ALSO
    ToolTypes and the Commodities Environment (in Reference Manual)
    EnqueueCxObj()
```

```
commodities.library/SetFilter                   commodities.library/SetFilter

   NAME
     SetFilter() (V36)

   SYNOPSIS
     SetFilter(filter, text);

     void SetFilter(CxObj *,IX *);

   FUNCTION
     Changes the matching condition of a Commodities input filter to
     that described by the input description string 'text.'

   DIAGNOSTICS
     The internal error of 'filter' will have the COERR_BADFILTER
     bit set or cleared depending on the failure or success (resp.)
     of SetFilter().

   SEE ALSO
     SetFilter(), SetFilterIX(), CxObjError()
     Commodities Input Messages and Filters
     Input Expressions and Description Strings
```

```
commodities.library/SetFilterIX               commodities.library/SetFilterIX

   NAME
     SetFilterIX() (V36)

   SYNOPSIS
     error = SetFilterIX(filter, ix);

     void SetFilterIX(CxObj *,IX *);

   FUNCTION
     Changes the matching condition of a Commodities input filter to
     that described by the binary input expression pointed by 'ix.'

     Input expressions are defined in the file <libraries/commodities.h>
     It is important to remember that the first field of the input
     expression structure must indicate which type of version of the
     input expression structure is being used.
     See <libraries/commodities.h>

   DIAGNOSTICS
     If 'error' returned is non-zero, it contains an error code from
     <libraries/commodities.h>.

   SEE ALSO
     SetFilter(), SetFilterIX(), CxObjError()
     Commodities Input Messages and Filters
     Input Expressions and Description Strings
```

```
commodities.library/SetTranslate             commodities.library/SetTranslate

   NAME
     SetTranslate() (V36)

   SYNOPSIS
     SetTranslate(translator, ie);

     void SetTranslate(CxObj *,IX *);

   FUNCTION
     This function replaces the translation list of a Commodities
     translate object 'translator' with the linked list starting at 'ie.'

     A null value for 'ie' indicates that the object 'translator'
     should swallow all Commodities messages that are sent its way.

     Note that the input events are not copied into Commodities private
     memory, but the value of ie is used--asynchronously to the
     application program--to find a chain of InputEvents in the
     application's data space.

     At the time of translation, though, each input event is copied
     into its own new Commodities message.

     This means that no other Commodities user, nor Commodities itself
     will be modifying your list of InputEvents.  On the other
     hand, your program must not corrupt the input event chain
     that has been presented to a translator.

   DIAGNOSTICS
     Returns zero if there is no problem, otherwise an error code
     with values found in <libraries/commodities.h>. The only forseeable
     error could occur if 'translator' was not in fact a translate object.

   SEE ALSO
     include:devices/inputevent.h
     CxTranslate()
```

cx_lib.library

Page 1

TABLE OF CONTENTS

cx.lib/ArgArrayDone
cx.lib/ArgArrayInit
cx.lib/ArgInt
cx.lib/ArgString
cx.lib/CxCustom
cx.lib/CxDebug
cx.lib/CxFilter
cx.lib/CxSender
cx.lib/CxSignal
cx.lib/CxTranslate
cx.lib/CxTypeFilter
cx.lib/FreeIEvents
cx.lib/HotKey
cx.lib/InvertString

cx_lib.library

Page 2

```
cx.lib/ArgArrayDone                                          cx.lib/ArgArrayDone

  NAME   ArgArrayDone()      ---    SCANNED LIBRARY (V36)

    ArgArrayDone();

    void ArgArrayDone(void);

  DESCRIPTION
    This function frees memory and does cleanup required
    by ArgArrayInit().  Don't call this until you are done using
    the ToolTypes argument strings.

  DIAGNOSTICS
    None.

  SEE ALSO
    ArgArrayInit()
```

cx_lib.library Page 3

```
cx.lib/ArgArrayInit                                         cx.lib/ArgArrayInit

  NAME   ArgArrayInit   --   SCANNED LIBRARY (V36)

    ttypes = ArgArrayInit(argc, argv);

    char **ArgArrayInit(int,char **);

  DESCRIPTION
    This function is not part of "commodities.library," but is
    in "amiga.lib."

    ArgArrayInit() returns a null-terminated array
    of strings suitable for sending to the icon.library function
    FindToolType().  This array will be the ToolTypes array of the
    program's icon, if it was started from the workbench.  It will
    be just 'argv' if the program was started from the CLI.

    Pass ArgArrayInit() your startup arguments passed to main().

    ArgArrayInit() requires that the icon.library be open
    (even if the caller was started from a CLI, so that the function
    FindToolType() can be used) and may call GetDiskObject(), so clean
    up is necessary when the strings are no longer needed.  The function
    ArgArrayDone() does just that.

    IMPORTANT: Your program must open icon.library and set up IconBase
               before calling this routine. In addition IconBase must
               remain valid until after ArgArrayDone() has been called!

    Use of these routines facilitates the use of ToolTypes or command
    line arguments to control end-user parameters in Commodities
    applications.  For example, a filter used to trap a keystroke for
    popping up a window might be created by something like this:

    char   *ttypes  = ArgArrayInit(argc, argv);
    CxObj   *filter = UserFilter(ttypes, "POPWINDOW", "alt f1");

    ... with ...

    CxObj   *
    UserFilter(tt, action_name, default_descr)
    char   **tt;             \* null-terminated (char *(*)[])   *\
    char   *action_name;     \* name of your semantic action      *\
    char   *default_descr;   \* used if user doesn't provide      *\
    {
       char   *desc;

       desc = FindToolType(tt, action_name);

       return ( CxFilter(desc? desc: default_descr) );
    }

    In this way the user can assign "alt f2" to the action by
    entering a tooltype in the programs icon of the form:

       POPWINDOW=alt f2

    or by starting the program from the CLI with like so:

       myprogram "POPWINDOW=alt f2"

  DIAGNOSTICS
    ArgArrayInit() will return NULL if any problems occur.

  SEE ALSO
```

cx_lib.library Page 4

```
    ArgArrayDone()
    ArgString()
    ArgInt()
    icon.library/FindToolType()
```

intuition.library Page 141

```
intuition.library/WindowLimits                     intuition.library/WindowLimits

    NAME
        WindowLimits -- Set the minimum and maximum limits of a window.

    SYNOPSIS
        Success = WindowLimits( Window, MinWidth, MinHeight, MaxWidth,
        D0                      A0      D0        D1         D2
                                MaxHeight )
                                D3

        BOOL WindowLimits( struct Window *, WORD, WORD, UWORD, UWORD );

    FUNCTION
        Sets the minimum and maximum limits of the window's size.  Until this
        routine is called, the window's size limits are equal to the initial
        values established in the OpenWindow() function.

        After a call to this routine, the Window will be able to be sized
        to any dimensions within the specified limits.

        If you don't want to change any one of the dimensions, set the limit
        argument for that dimension to zero.  If any of the limit arguments
        is equal to zero, that argument is ignored and the initial setting
        of that parameter remains undisturbed.

        If any of the arguments is out of range (minimums greater than the
        current size, maximums less than the current size), that limit
        will be ignored, though the others will still take effect if they
        are in range.  If any are out of range, the return value from this
        procedure will be FALSE.  If all arguments are valid, the return
        value will be TRUE.

        If you want your window to be able to become "as large as possible"
        you may put -1 (i.e. ~0) in either or both Max arguments.  But
        please note: screen sizes may vary for several reasons, and you
        must be able to handle any possible size of window you might end
        up with if you use this method.  Note that you can use the function
        GetScreenData() to find out how big the screen your window appears in
        is.  That function is particularly useful if your window is in
        the Workbench screen.  You may also refer to the WScreen field
        in your window structure, providing that your window remains open,
        which will ensure that the screen remains open, and thus the
        pointer remains valid.

        If the user is currently sizing this window, the new limits will
        not take effect until after the sizing is completed.

    INPUTS
        Window = pointer to a Window structure
        MinWidth, MinHeight, MaxWidth, MaxHeight = the new limits for the size
            of this window.  If any of these is set to zero, it will
            be ignored and that setting will be unchanged.

    RESULT
        Returns TRUE if everything was in order.  If any of the parameters was
        out of range (minimums greater than current size, maximums less than
        current size), FALSE is returned and the errant limit request is
        not fulfilled (though the valid ones will be).

    BUGS

    SEE ALSO
        GetScreenData()
```

intuition.library Page 142

```
intuition.library/WindowToBack                     intuition.library/WindowToBack

    NAME
        WindowToBack -- Ask Intuition to send a window behind others.

    SYNOPSIS
        WindowToBack( Window )
                      A0

        VOID WindowToBack( struct Window * );

    FUNCTION
        This routine sends a request to Intuition asking to send the window
        in back of all other windows in the screen.

        Note that the window will not be depth-arranged immediately, but rather
        will be arranged the next time Intuition receives an input event,
        which happens currently at a minimum rate of ten times per second,
        and a maximum of sixty times a second.

        Remember that WFLG_BACKDROP windows cannot be depth-arranged.

    INPUTS
        Window = pointer to the structure of the window to be sent to the back

    RESULT
        None

    BUGS

    SEE ALSO
        MoveWindow(), SizeWindow(), WindowToFront(), MoveWindowInFrontOf()
```

cx_lib.library Page 7

```
cx.lib/CxCustom                                           cx.lib/CxCustom

  NAME   CxCustom()   --   MACRO (V36)

    custom = CxCustom(action, id);

    CxObj *CxCustom(LONG(*)(),LONG);
   LONG   (*action)();

  DESCRIPTION
    This function creates a custom Commodities object.  The action
    of this object on receiving a Commodities message is to
    call a function of the application programmer's choice.

    The function provided ('action') will be passed a pointer to
    the actual commodities message (in commodities private data
    space), and will actually execute as part of the input handler
    system task.  Among other things, the value of 'id' can be
    recovered from the message by using the function CxMsgID().

    The purpose of this function is two-fold.  First, it allows
    programmers to create Commodities Exchange objects with
    functionality that was not imagined or chosen for inclusion
    by the designers.  Secondly, this is the only way to act
    synchronously with Commodities.

    For further explanation and examples, consult the Reference
    Manual.

    This function is a C-language macro for CreateCxObj(), defined
    in <libraries/commodities.h>.

  DIAGNOSTICS
    Returns NULL if the custom object cannot be created.

  SEE ALSO
    CreateCxObj()
    Custom Objects (in Reference Manual)
    CxMsgID()
```

cx_lib.library Page 8

```
cx.lib/CxDebug                                             cx.lib/CxDebug

  NAME   CxDebug()   --   MACRO (V36)

    debugger = CxDebug(id);

    CxObj *CxDebug(LONG);

  DESCRIPTION
    This function creates a Commodities debug object.  The action
    of this object on receiving a Commodities message is to
    print out information about the Commodities message through
    the Serial port (using the kprintf() routine).  The value
    'id' will also be displayed.

    Note that this is a synchronous occurence (the printing
    is done by the input device task).  If screen or file
    output is desired, using a sender object instead of
    debug object is necessary, since such output is best
    done by your application process.

    This function is a C-language macro for CreateCxObj(), defined
    in <libraries/commodities.h>.

  DIAGNOSTICS
    Returns NULL if the debug object cannot be created.

  SEE ALSO
    CreateCxObj()
    CxSender()
    exec_support/kprintf()
```

```
cx.lib/CxFilter                                         cx.lib/CxFilter

  NAME   CxFilter()   --   MACRO (V36)

    filter = CxFilter(description);

    CxObj *CxFilter(BYTE *);

  DESCRIPTION
    Creates an input event filter object which matches Commodities
    Input Messages fitting the 'description' string.  If
    'description' is NULL, the filter will not match any messages.

    A filter may be modified by the functions SetFilter(), using
    a description string, and SetFilterIX(), which takes a
    binary Input Expression as a parameter.

    This function is a C-language macro for CreateCxObj(), defined
    in <libraries/commodities.h>.

  DIAGNOSTICS
    Returns NULL if the function fails, which only occurs
    if there is no memory for the new filter object.  If there
    is a problem in the description string, the internal error
    code of the filter object will be set to so indicate.  This
    error code may be interrogated using the function CxObjError().

  SEE ALSO
    CreateCxObj()
    SetFilter(), SetFilterIX(), CxObjError()
    Input Expressions and Description Strings
    Objects and Messages (in Reference Manual)
    Error Handling   (in Reference Manual)
```

```
cx.lib/CxSender                                         cx.lib/CxSender

  NAME   CxSender()   --   MACRO (V36)

    sender = CxSender(port, id)

    CxObj *CxSender(struct MsgPort *,LONG);

  DESCRIPTION
    This function creates a Commodities sender object.  The action
    of this object on receiving a Commodities message is to
    copy the Commodities message into a standard Exec Message,
    to put the value 'id' in the message as well, and to send the
    message off to the Message Port 'port.'

    The value 'id' is used so that an application can monitor
    messages from several senders at a single port.  It can
    be retrieved from the Exec message by using the function
    CxMsgID(). The value can be a simple integer ID, or a pointer
    to some application data structure, for example.

    Note that Exec messages sent by sender objects arrive
    asynchronously at the destination port.  Do not assume
    anything about the status of the Commodities message which
    was copied into the Exec message you received.

    All Exec messages sent to your ports must be replied.
    Messages may be replied after the sender object has been
    deleted.

    This function is a C-language macro for CreateCxObj(), defined
    in <libraries/commodities.h>.

  DIAGNOSTICS
    Returns NULL if the sender object cannot be created.

  SEE ALSO
    CreateCxObj()
    exec.library/PutMsg(), exec.library/ReplyMsg()
    CxMsgID()
```

cx_lib.library

Page 11

```
cx.lib/CxSignal                                                  cx.lib/CxSignal

  NAME   CxSignal()   --   MACRO (V36)

    signaler = CxSignal(task, signal);

    CxObj *CxSignal(struct Task *,LONG);

  DESCRIPTION
    This function creates a Commodities signal object.  The action
    of this object on receiving a Commodities message is to
    send the 'signal' to the 'task.'  The caller is responsible
    for allocating the signal and determining the proper task ID.

    Note that 'signal' is the signal value as returned by
    AllocSignal() (example: 3) , not the mask made from
    that value (i.e., not binary 0000000000001000).

    This function is a C-language macro for CreateCxObj(), defined
    in <libraries/commodities.h>.

  DIAGNOSTICS
    Returns NULL if the object could not be created.

  SEE ALSO
    CreateCxObj()
    Objects and Messages (in Reference Manual)
    exec.library/Signal(), exec.library/AllocSignal(),
    exec.library/FindTask()
```

cx_lib.library

Page 12

```
cx.lib/CxTranslate                                            cx.lib/CxTranslate

  NAME   CxTranslate()   --   MACRO (V36)

    translator = CxTranslate(ie);

    CxObj *CxTranslate(struct InputEvent *);

  DESCRIPTION
    This function creates a Commodities 'translate' object.
    The action of this object on receiving a Commodities
    message is to replace that message in the commodities network
    with a chain of Commodities input messages.

    One new Commodities input message for each input event
    in the linked list starting at 'ie' (and NULL terminated).
    The routing information of the new input messages will be
    copied from the input message they replace.

    The linked list of input events associated with a translate
    object can be changed using the SetTranslate() function.

    If 'ie' is NULL, the null translation occurs: that is, the
    original commodities input message is disposed, and no
    others are created to take its place.

    This function is a C-language macro for CreateCxObj(), defined
    in <libraries/commodities.h>.

  DIAGNOSTICS
    Returns NULL if the translate object cannot be created.

  SEE ALSO
    CreateCxObj()
    SetTranslate()
    Commodities Input Messages and Filters
```

```
cx.lib/CxTypeFilter                                         cx.lib/CxTypeFilter

  NAME   CxTypeFilter()   --   MACRO (V36)

    typef = CxTypeFilter(typemask);

    CxObj *CxTypeFilter(LONG);

  DESCRIPTION
    Creates a Commodities Object similar to CxFilter(), but one
    that diverts all Commodities messages whose type, which is
    always a power of two, matches a bit set in 'typemask.'

    Values of message types are given in <libraries/commodities.h>.

    This function is a C-language macro for CreateCxObj(), defined
    in <libraries/commodities.h>.

  DIAGNOSTICS
    Returns NULL if the function fails, which only occurs
    if there is no memory for the new filter object.  If there
    is a problem in the description string, the internal error
    code of the filter object will be set to so indicate.  This
    error code may be interrogated using the function CxObjError().

  SEE ALSO
    CreateCxObj()
    SetFilter(), SetFilterIX(), CxObjError()
    Object and Messages (in Reference Manual)
    Input Expressions and Description Strings
    Error Handling
```

```
cx.lib/FreeIEvents                                           cx.lib/FreeIEvents

  NAME   FreeIEvents()   --   SCANNED LIBRARY (V36)

    FreeIEvents(ie)

    void FreeIEvents(struct InputEvent *ie);

  DESCRIPTION
    This function frees a linked list of input events returned by
    InvertString().

  DIAGNOSTICS
    None.

  SEE ALSO
    ParseIX()
    InvertString()
```

cx_lib.library Page 15

```
cx.lib/HotKey                                                   cx.lib/HotKey

  NAME   HotKey()   --   SCANNED LIBRARY (V36)

    filter = Hotkey(descr, port, ID);

    CxObj *HotKey(char *,struct MsgPort *,LONG);

  DESCRIPTION
    This function is not part of "commodities.library," but
    is in the scanned library amiga.lib.  It creates
    a triad of Commodities objects to accomplish a high-level
    function.

    The three objects are a filter, which is created to match
    by the call CxFilter(descr), a sender created by the call
    CxSender(port, ID), and a translator which is created by
    CxTranslate(NULL), so that it swallows any Commodities
    input event messages that are passed down by the filter.

    This is the simple way to get a message sent to your program
    when the user performs a particular input action.

    It is strongly recommended that the ToolTypes environment
    be used to allow the user to specify the input descriptions
    for your application's hotkeys.

  DIAGNOSTICS
    Returns NULL and cleans up after itself if any problems occur
    creating the objects.  It may be wise to test filter using
    CxObjError() to insure that 'descr' was a valid description.

  SEE ALSO
    ToolTypes and the Commodities Environment (in Reference Manual)
    CxFilter(), CxSender(), CxTranslate(), CxObjError()
```

cx_lib.library Page 16

```
cx.lib/InvertString                                       cx.lib/InvertString

  NAME   InvertString()   --   SCANNED LIBRARY (V36)

    event =InvertString(str, km)

    struct InputEvent *InvertString(UBYTE *,CPTR *);

  DESCRIPTION
    This function returns a linked list of input events which would
    translate into the string using the keymap 'km' (of the system
    default keymap if 'km' is NULL).

    The null-terminated 'str' may contain:
    -ANSI character codes
    -backslash escaped characters:
       \n   -   return
       \r   -   return
       \t   -   tab
       \0   -   don't use this, ok?
       \\   -   backslash
    -a text description of an input event as used by ParseIX(),
     enclosed in angle brackets.

    An example is:
       abc<alt f1>\nhi there.

    NOTE: you are responsible for freeing the InputEvents that this
    function allocates.  You may use FreeIEvents().

  DIAGNOSTICS
    Returns NULL if there is a problem, most often an illegal description
    enclosed in angles.

  SEE ALSO
    ParseIX()
    FreeIEvents()
```

diskfont.library

TABLE OF CONTENTS

diskfont.library/AvailFonts
diskfont.library/DisposeFontContents
diskfont.library/NewFontContents
diskfont.library/NewScaledDiskFont
diskfont.library/OpenDiskFont

diskfont.library

```
diskfont.library/AvailFonts                        diskfont.library/AvailFonts

   NAME
       AvailFonts -- Inquire available memory & disk fonts.

   SYNOPSIS
       error = AvailFonts(buffer, bufBytes, flags);
                          A0      D0        D1

       LONG AvailFonts( struct AvailFontsHeader *buffer, LONG bufBytes,
               ULONG flags );

   FUNCTION
       AvailFonts fills a user supplied buffer with the structure,
       described below, that contains information about all the
       fonts available in memory and/or on disk.  Those fonts
       available on disk need to be loaded into memory and opened
       via OpenDiskFont, those already in memory are accessed via
       OpenFont.  The TextAttr structure required by the open calls
       is part of the information AvailFonts supplies.

       When AvailFonts fails, it returns the number of extra bytes
       it needed to complete the command.  Add this number to your
       current buffer size, allocate a new buffer, and try again.

   INPUTS
       buffer - memory to be filled with struct AvailFontsHeader
               followed by an array of AvailFonts elements, which
               contains entries for the available fonts and their
               names.

       bufBytes - the number of bytes in the buffer
       flags - AFF_MEMORY is set to search memory for fonts to fill
               the structure, AFF_DISK is set to search the disk for
               fonts to fill the structure.  AFF_SCALED is set to
               not filter out memory fonts that are not designed.
               Any combination may be specified.  AFF_TAGGED is set
               to fill the buffer with TAvailFonts elements instead
               of AvailFonts elements.

   RESULTS
       buffer - filled with struct AvailFontsHeader followed by the
               [T]AvailFonts elements, There will be duplicate entries
               for fonts found both in memory and on disk, differing
               only by type.  The existence of a disk font in the
               buffer indicates that it exists as an entry in a font
               contents file -- the underlying font file has not been
               checked for validity, thus an OpenDiskFont of it may
               fail.
       error - if non-zero, this indicates the number of bytes needed
               for AvailFonts in addition to those supplied.  Thus
               structure elements were not returned because of
               insufficient bufBytes.

   EXAMPLE
       int afShortage, afSize;
       struct AvailFontsHeader *afh;

       ...

       afSize = 400;
       do {
           afh = (struct AvailFontsHeader *) AllocMem(afSize, 0);
           if (afh) {
               afShortage = AvailFonts(afh, afSize, AFF_MEMORY|AFF_DISK);
               if (afShortage) {
                FreeMem(afh, afSize);
                afSize += afShortage;
            }
        }
        else {
            fail("AllocMem of AvailFonts buffer afh failed\n");
            break;
        }
}
        while (afShortage);

    /*
     * if (afh) non-zero here, then:
     * 1. it points to a valid AvailFontsHeader
     * 2. it must have FreeMem(afh, afSize) called for it after use
     */
```

```
diskfont.library/DisposeFontContents     diskfont.library/DisposeFontContents

   NAME
       DisposeFontContents -- Free the result from NewFontContents. (V34)

   SYNOPSIS
       DisposeFontContents(fontContentsHeader)
                           A1

       VOID DisposeFontContents( struct FontContentsHeader * );

   FUNCTION
       This function frees the array of FontContents entries
       returned by NewFontContents.

   INPUTS
       fontContentsHeader - a struct FontContentsHeader pointer
           returned by NewFontContents.

   EXCEPTIONS
       This command was first made available as of version 34.

       A fontContentsHeader other than one acquired by a call
       NewFontContents will crash.

   SEE ALSO
       NewFontContents to get structure freed here.
```

```
diskfont.library/NewFontContents              diskfont.library/NewFontContents

   NAME
       NewFontContents -- Create a FontContents image for a font. (V34)

   SYNOPSIS
       fontContentsHeader = NewFontContents(fontsLock,fontName)
      D0                                  A0        A1

       struct FontContentsHeader *NewFontContents( BPTR, char * );

   FUNCTION
       This function creates a new array of FontContents entries
       that describe all the fonts associated with the fontName,
       specifically, all those in the font directory whose name
       is that of the font sans the ".font" suffix.

   INPUTS
       fontsLock - a DOS lock on the FONTS: directory (or other
           directory where the font contents file and associated
           font directory resides).
       fontName - the font name, with the ".font" suffix, which
           is also the name of the font contents file.

   RESULT
       fontContentsHeader - a struct FontContentsHeader pointer.

   EXCEPTIONS
       This command was first made available as of version 34.

       D0 is zero if the fontName is does not have a ".font" suffix,
       if the fontName is too long, if a DOS error occurred, or if
       memory could not be allocated for the fontContentsHeader.

   SEE ALSO
       DisposeFontContents to free the structure acquired here.
```

```
diskfont.library/NewScaledDiskFont          diskfont.library/NewScaledDiskFont

   NAME
       NewScaledDiskFont -- Create a DiskFont scaled from another. (V36)

   SYNOPSIS
       header = NewScaledDiskFont(srcFont, destTextAttr)
       D0                         A0       A1

       struct DiskFontHeader *NewScaledDiskFont( struct TextFont *,
               struct TTextAttr * );

   INPUTS
       srcFont - the font from which the scaled font is to be
           constructed.
       destTextAttr - the desired attributes for the new scaled
           font.  This may be a structure of type TextAttr or
           TTextAttr.

   RESULT
       header - a pointer to a DiskFontHeader structure.  This is not
               being managed by the diskfont.library, however.

   NOTES
           o   This function may use the blitter.
           o   Fonts containing characters that render wholly outside
               the character advance cell are currently not scalable.
           o   The font, and memory allocated for the scaled font can
               can be freed by calling StripFont() on the font,
               and then calling UnLoadSeg() on the segment created
               by this function.

               Both the TextFont structure, and segment pointer are contained
               within the DiskFontHeader struct.  The DiskFontHeader structure
               will also be freed as part of the UnLoadSeg() call.
               StripFont() is a new graphics.library call as of V36.
```

```
diskfont.library/OpenDiskFont                    diskfont.library/OpenDiskFont

   NAME
       OpenDiskFont - load and get a pointer to a disk font.

   SYNOPSIS
       font = OpenDiskFont(textAttr)
       D0                  A0

   FUNCTION
       This function finds the font with the specified textAttr on
       disk, loads it into memory, and returns a pointer to the font
       that can be used in subsequent SetFont and CloseFont calls.
       It is important to match this call with a corresponding
       CloseFont call for effective management of font memory.

       If the font is already in memory, the copy in memory is used.
       The disk copy is not reloaded.

   INPUTS
       textAttr - a TextAttr structure that describes the text font
               attributes desired.

   RESULTS
       D0 is zero if the desired font cannot be found.

   NOTES
       As of V36, OpenDiskFont() will automatically attempt to
       construct a font for you if:

               You have requested a font size which does not exist
               as a designed font, and

               You have not set the DESIGNED bit in the ta_Flags
               field of the TextAttr, or TTextAttr struct.

       Constructed fonts are created by scaling a designed font.
       A designed font is one which typically resides on disk,
       or in ROM (e.g., a font which has been designed by hand
       using a drawing tool).  Designed fonts generally look better
       than fonts constructed by the font scaler, but designed
       fonts also require disk space for each font size.

       Always set the DESIGNED bit if you do not want constructed fonts,
       or use AvailFonts() to find out which font sizes already exist.

   BUGS
       This routine will not work well with font names whose file
       name components are longer than the maximum allowed
       (30 characters).
```

TABLE OF CONTENTS

dos.library/AbortPkt
dos.library/AddBuffers
dos.library/AddDosEntry
dos.library/AddPart
dos.library/AddSegment
dos.library/AllocDosObject
dos.library/AssignAdd
dos.library/AssignLate
dos.library/AssignLock
dos.library/AssignPath
dos.library/AttemptLockDosList
dos.library/ChangeMode
dos.library/CheckSignal
dos.library/Cli
dos.library/Close
dos.library/CompareDates
dos.library/CreateDir
dos.library/CreateNewProc
dos.library/CreateProc
dos.library/CurrentDir
dos.library/DateStamp
dos.library/DateToStr
dos.library/Delay
dos.library/DeleteFile
dos.library/DeleteVar
dos.library/DeviceProc
dos.library/DoPkt
dos.library/DupLock
dos.library/DupLockFromFH
dos.library/EndNotify
dos.library/ErrorReport
dos.library/ExAll
dos.library/Examine
dos.library/ExamineFH
dos.library/Execute
dos.library/Exit
dos.library/ExNext
dos.library/Fault
dos.library/FGetC
dos.library/FGets
dos.library/FilePart
dos.library/FindArg
dos.library/FindCliProc
dos.library/FindDosEntry
dos.library/FindSegment
dos.library/FindVar
dos.library/Flush
dos.library/Format
dos.library/FPutC
dos.library/FPuts
dos.library/FRead
dos.library/FreeArgs
dos.library/FreeDeviceProc
dos.library/FreeDosEntry
dos.library/FreeDosObject
dos.library/FWrite
dos.library/GetArgStr
dos.library/GetConsoleTask
dos.library/GetCurrentDirName
dos.library/GetDeviceProc
dos.library/GetFileSysTask
dos.library/GetProgramDir
dos.library/GetProgramName
dos.library/GetPrompt

dos.library/GetVar
dos.library/Info
dos.library/Inhibit
dos.library/Input
dos.library/InternalLoadSeg
dos.library/InternalUnLoadSeg
dos.library/IoErr
dos.library/IsFileSystem
dos.library/IsInteractive
dos.library/LoadSeg
dos.library/Lock
dos.library/LockDosList
dos.library/LockRecord
dos.library/LockRecords
dos.library/MakeDosEntry
dos.library/MakeLink
dos.library/MatchEnd
dos.library/MatchFirst
dos.library/MatchNext
dos.library/MatchPattern
dos.library/MatchPatternNoCase
dos.library/MaxCli
dos.library/NameFromFH
dos.library/NameFromLock
dos.library/NewLoadSeg
dos.library/NextDosEntry
dos.library/Open
dos.library/OpenFromLock
dos.library/Output
dos.library/ParentDir
dos.library/ParentOfFH
dos.library/ParsePattern
dos.library/ParsePatternNoCase
dos.library/PathPart
dos.library/PrintFault
dos.library/PutStr
dos.library/Read
dos.library/ReadArgs
dos.library/ReadItem
dos.library/ReadLink
dos.library/Relabel
dos.library/RemAssignList
dos.library/RemDosEntry
dos.library/RemSegment
dos.library/Rename
dos.library/ReplyPkt
dos.library/RunCommand
dos.library/SameDevice
dos.library/SameLock
dos.library/Seek
dos.library/SelectInput
dos.library/SelectOutput
dos.library/SendPkt
dos.library/SetArgStr
dos.library/SetComment
dos.library/SetConsoleTask
dos.library/SetCurrentDirName
dos.library/SetFileDate
dos.library/SetFileSize
dos.library/SetFileSysTask
dos.library/SetIoErr
dos.library/SetMode
dos.library/SetProgramDir
dos.library/SetProgramName
dos.library/SetPrompt
dos.library/SetProtection

dos.library/SetVar
dos.library/SetVBuf
dos.library/SplitName
dos.library/StartNotify
dos.library/StrToDate
dos.library/StrToLong
dos.library/SystemTagList
dos.library/UnGetC
dos.library/UnLoadSeg
dos.library/UnLock
dos.library/UnLockDosList
dos.library/UnLockRecord
dos.library/UnLockRecords
dos.library/VFPrintf
dos.library/VFWritef
dos.library/VPrintf
dos.library/WaitForChar
dos.library/WaitPkt
dos.library/Write
dos.library/WriteChars

```
dos.library/AbortPkt                                     dos.library/AbortPkt

   NAME
       AbortPkt -- Aborts an asynchronous packet, if possible. (V36)

   SYNOPSIS
       AbortPkt(port, pkt)
                D1    D2

       void AbortPkt(struct MsgPort *, struct DosPacket *)

   FUNCTION
       This attempts to abort a packet sent earlier with SendPkt to a
       handler.  There is no guarantee that any given handler will allow
       a packet to be aborted, or if it is aborted whether function
       requested completed first or completely.  After calling AbortPkt(),
       you must wait for the packet to return before reusing it or
       deallocating it.

   INPUTS
       port - port the packet was sent to
       pkt  - the packet you wish aborted

   BUGS
       As of V37, this function does nothing.

   SEE ALSO
       SendPkt(), DoPkt(), WaitPkt()
```

```
dos.library/AddBuffers                                  dos.library/AddBuffers

   NAME
        AddBuffers -- Changes the number of buffers for a filesystem (V36)

   SYNOPSIS
        success = AddBuffers(filesystem, number)
        D0                   D1          D2

        BOOL AddBuffers(char *, LONG)

   FUNCTION
        Adds buffers to a filesystem.  If it succeeds, the number of current
        buffers is returned in IoErr().  Note that "number" may be negative.
        The amount of memory used per buffer, and any limits on the number of
        buffers, are dependant on the filesystem in question.
        If the call succeeds, the number of buffers in use on the filesystem
        will be returned by IoErr().

   INPUTS
        filesystem - Name of device to add buffers to (with ':').
        number     - Number of buffers to add.  May be negative.

   RESULT
        success    - Success or failure of command.

   BUGS
        The V36 ROM filesystem (FFS/OFS) doesn't return the right number of
        buffers unless preceded by an AddBuffers(fs,-1) (in-use buffers aren't
        counted).  This is fixed in V37.

   SEE ALSO
```

```
dos.library/AddDosEntry                                dos.library/AddDosEntry

   NAME
        AddDosEntry -- Add a Dos List entry to the lists (V36)

   SYNOPSIS
        success = AddDosEntry(dlist)
        D0                    D1

        BOOL AddDosEntry(struct DosList *)

   FUNCTION
        Adds a device, volume or assign to the dos devicelist.  Can fail if it
        conflicts with an existing entry (such as another assign to the same
        name or another device of the same name).  Volume nodes with different
        dates and the same name CAN be added, or with names that conflict with
        devices or assigns.  Note: the dos list does NOT have to be locked to
        call this.  Do not access dlist after adding unless you have locked the
        Dos Device list.

   INPUTS
        dlist   - Device list entry to be added.

   RESULT
        success - Success/Failure indicator

   SEE ALSO
        RemDosEntry(), FindDosEntry(), NextDosEntry(), LockDosList(),
        MakeDosEntry(), FreeDosEntry()
```

```
dos.library/AddPart                                       dos.library/AddPart

   NAME
       AddPart -- Appends a file/dir to the end of a path (V36)

   SYNOPSIS
       success = AddPart( dirname, filename, size )
       D0                 D1       D2        D3

       BOOL AddPart( UBYTE *, UBYTE *, ULONG )

   FUNCTION
       This function adds a file, directory, or subpath name to a directory
       path name taking into account any required separator characters.  If
       filename is a fully-qualified path it will totally replace the current
       value of dirname.

   INPUTS
       dirname  - the path to add a file/directory name to.
       filename - the filename or directory name to add.  May be a relative
                  pathname from the current directory (example: foo/bar).
                  Can deal with leading '/'(s), indicating one directory up
                  per '/', or with a ':', indicating it's relative to the
                  root of the appropriate volume.
       size     - size in bytes of the space allocated for dirname.  Must
                  not be 0.

   RESULT
       success - non-zero for ok, FALSE if the buffer would have overflowed.
                 If an overflow would have occured, dirname will not be
                 changed.

   BUGS
       Doesn't check if a subpath is legal (i.e. doesn't check for ':'s) and
       doesn't handle leading '/'s in 2.0 through 2.02 (V36).  V37 fixes
       this, allowing filename to be any path, including absolute.

   SEE ALSO
       Filepart(), PathPart()
```

```
dos.library/AddSegment                                 dos.library/AddSegment

   NAME
       AddSegment - Adds a resident segment to the resident list (V36)

   SYNOPSIS
       success = AddSegment(name, seglist, type)
       D0                   D1    D2       D3

       BOOL AddSegment(char *, BPTR, LONG)

   FUNCTION
       Adds a segment to the Dos resident list, with the specified Seglist
       and type (stored in seg_UC - normally 0).  NOTE: currently unused
       types may cause it to interpret other registers (d4-?) as additional
       parameters in the future.

       Do NOT build Segment structures yourself!

   INPUTS
       name    - name for the segment
       seglist - Dos seglist of code for segment
       type    - initial usecount, normally 0

   RESULT
       success - success or failure

   BUGS

   SEE ALSO
       FindSegment(), RemSegment(), LoadSeg()
```

```
dos.library/AllocDosObject                    dos.library/AllocDosObject

   NAME
       AllocDosObject -- Creates a dos object (V36)

   SYNOPSIS
       ptr = AllocDosObject(type, tags)
       D0                   D1    D2

       void *AllocDosObject(ULONG, struct TagItem *)

       ptr = AllocDosObjectTagList(type, tags)
       D0                          D1    D2

       void *AllocDosObjectTagList(ULONG, struct TagItem *)

       ptr = AllocDosObjectTags(type, Tag1, ...)

       void *AllocDosObjectTags(ULONG, ULONG, ...)

   FUNCTION
       Create one of several dos objects, initializes it, and returns it
       to you.  Note the DOS_STDPKT returns a pointer to the sp_Pkt of the
       structure.

   INPUTS
       type - type of object requested
       tags - pointer to taglist with additional information

   RESULT
       packet - pointer to the object or NULL

   SEE ALSO
       FreeDosObject(), <dos/dostags.h>, <dos/dos.h>
```

```
dos.library/AssignAdd                              dos.library/AssignAdd

   NAME
       AssignAdd -- Adds a lock to an assign for multi-directory assigns (V36)

   SYNOPSIS
       success = AssignAdd(name,lock)
       D0                  D1   D2

       BOOL AssignAdd(char *,BPTR)

   FUNCTION
       Adds a lock to an assign, making or adding to a multi-directory
       assign.  Note that this only will succeed on an assign created with
       AssignLock(), or an assign created with AssignLate() which has been
       resolved (converted into a AssignLock()-assign).

       NOTE: you should not use the lock in any way after making this call
       successfully.  It becomes the part of the assign, and will be unlocked
       by the system when the assign is removed.  If you need to keep the
       lock, pass a lock from DupLock() to AssignLock().

   INPUTS
       name - Name of device to assign lock to (without trailing ':')
       lock - Lock associated with the assigned name

   RESULT
       success - Success/failure indicator.  On failure, the lock is not
                 unlocked.

   SEE ALSO
       Lock(), AssignLock(), AssignPath(), AssignLate(), DupLock(),
       RemAssignList()
```

dos.library Page 11

```
dos.library/AssignLate                                 dos.library/AssignLate

   NAME
        AssignLate -- Creates an assignment to a specified path later (V36)

   SYNOPSIS
        success = AssignLate(name,path)
        D0                   D1   D2

        BOOL AssignLate(char *,char *)

   FUNCTION
        Sets up a assignment that is expanded upon the FIRST reference to the
        name.  The path (a string) would be attached to the node.  When
        the name is referenced (Open("FOO:xyzzy"...), the string will be used
        to determine where to set the assign to, and if the directory can be
        locked, the assign will act from that point on as if it had been
        created by AssignLock().

        A major advantage is assigning things to unmounted volumes, which
        will be requested upon access (useful in startup sequences).

   INPUTS
        name - Name of device to be assigned (without trailing ':')
        path - Name of late assignment to be resolved on the first reference.

   RESULT
        success - Success/failure indicator of the operation

   SEE ALSO
        Lock(), AssignAdd(), AssignPath(), AssignLock(),
```

dos.library Page 12

```
dos.library/AssignLock                                 dos.library/AssignLock

   NAME
        AssignLock -- Creates an assignment to a locked object (V36)

   SYNOPSIS
        success = AssignLock(name,lock)
        D0                   D1   D2

        BOOL AssignLock(char *,BPTR)

   FUNCTION
        Sets up an assign of a name to a given lock.  Passing NULL for a lock
        cancels any outstanding assign to that name.  If an assign entry of
        that name is already on the list, this routine replaces that entry.  If
        an entry is on the list that conflicts with the new assign, then a
        failure code is returned.

        NOTE: you should not use the lock in any way after making this call
        successfully.  It becomes the assign, and will be unlocked by the
        system when the assign is removed.  If you need to keep the lock,
        pass a lock from DupLock() to AssignLock().

   INPUTS
        name - Name of device to assign lock to (without trailing ':')
        lock - Lock associated with the assigned name

   RESULT
        success - Success/failure indicator.  On failure, the lock is not
                  unlocked.

   SEE ALSO
        Lock(), AssignAdd(), AssignPath(), AssignLate(), DupLock(),
        RemAssignList()
```

```
dos.library/AssignPath                                   dos.library/AssignPath

   NAME
       AssignPath -- Creates an assignment to a specified path (V36)

   SYNOPSIS
       success = AssignPath(name,path)
       D0                   D1   D2

       BOOL AssignPath(char *,char *)

   FUNCTION
       Sets up a assignment that is expanded upon EACH reference to the name.
       This is implemented through a new device list type (DLT_ASSIGNPATH, or
       some such).  The path (a string) would be attached to the node.  When
       the name is referenced (Open("FOO:xyzzy"...), the string will be used
       to determine where to do the open.  No permanent lock will be part of
       it.  For example, you could AssignPath() c2: to df2:c, and references
       to c2: would go to df2:c, even if you change disks.

       The other major advantage is assigning things to unmounted volumes,
       which will be requested upon access (useful in startup sequences).

   INPUTS
       name - Name of device to be assigned (without trailing ':')
       path - Name of late assignment to be resolved at each reference

   RESULT
       success - Success/failure indicator of the operation

   SEE ALSO
       AssignAdd(), AssignLock(), AssignLate(), Open()
```

```
dos.library/AttemptLockDosList                   dos.library/AttemptLockDosList

   NAME
       AttemptLockDosList -- Attempt to lock the Dos Lists for use (V36)

   SYNOPSIS
       dlist = AttemptLockDosList(flags)
       D0                         D1

       struct DosList *AttemptLockDosList(ULONG)

   FUNCTION
       Locks the dos device list in preparation to walk the list.  If the
       list is 'busy' then this routine will return NULL.  See LockDosList()
       for more information.

   INPUTS
       flags - Flags stating which types of nodes you want to lock.

   RESULT
       dlist - Pointer to the beginning of the list or NULL.  Not a valid
               node!

   SEE ALSO
       LockDosList(), UnLockDosList(), Forbid(), NextDosEntry()
```

dos.library Page 15

```
dos.library/ChangeMode                                    dos.library/ChangeMode

   NAME
        ChangeMode - Change the current mode of a lock or filehandle (V36)

   SYNOPSIS
        success = ChangeMode(type, object, newmode)
        D0                    D1     D2      D3

        BOOL ChangeMode(ULONG, BPTR, ULONG)

   FUNCTION
        This allows you to attempt to change the mode in use by a lock or
        filehandle.  For example, you could attempt to turn a shared lock
        into an exclusive lock.  The handler may well reject this request.
        Warning: if you use the wrong type for the object, the system may
        crash.

   INPUTS
        type    - Either CHANGE_FH or CHANGE_LOCK
        object  - A lock or filehandle
        newmode - The new mode you want

   RESULT
        success - Boolean

   BUGS
        Did not work in 2.02 or before (V36).  Works in V37.  In the
        earlier versions, it can crash the machine.

   SEE ALSO
        Lock(), Open()
```

dos.library Page 16

```
dos.library/CheckSignal                                  dos.library/CheckSignal

   NAME
        CheckSignal -- Checks for break signals (V36)

   SYNOPSIS
        signals = CheckSignal(mask)
        D0                    D1

        ULONG CheckSignals(ULONG)

   FUNCTION
        This function checks to see if any signals specified in the mask have
        been set and if so, returns them.  Otherwise it returns FALSE.
        All signals specified in mask will be cleared.

   INPUTS
        mask    - Signals to check for.

   RESULT
        signals - Signals specified in mask that were set.

   SEE ALSO
```

```
dos.library/Cli                                              dos.library/Cli

   NAME
       Cli -- Returns a pointer to the CLI structure of the process (V36)

   SYNOPSIS
       cli_ptr = Cli()
       D0

       struct CommandLineInterface *Cli(void)

   FUNCTION
       Returns a pointer to the CLI structure of the current process, or NULL
       if the process has no CLI structure.

   RESULT
       cli_ptr - pointer to the CLI structure, or NULL.

   SEE ALSO
```

```
dos.library/Close                                          dos.library/Close

   NAME
       Close -- Close an open file

   SYNOPSIS
       success = Close( file )
          D0             D1

       BOOL Close(BPTR)

   FUNCTION
       The file specified by the file handle is closed. You must close all
       files you explicitly opened, but you must not close inherited file
       handles that are passed to you (each filehandle must be closed once
       and ONLY once).  If Close() fails, the file handle is still
       deallocated and should not be used.

   INPUTS
       file - BCPL pointer to a file handle

   RESULTS
       success - returns if Close() succeeded.  Note that it might fail
                 depending on buffering and whatever IO must be done to
                 close a file being written to.  NOTE: this return value
                 did not exist before V36!

   SEE ALSO
       Open(), OpenFromLock()
```

dos.library Page 19

```
dos.library/CompareDates                                   dos.library/CompareDates

   NAME
        CompareDates -- Compares two datestamps (V36)

   SYNOPSIS
        result = CompareDates(date1,date2)
        D0                     D1     D2

        LONG CompareDates(struct DateStamp *,struct DateStamp *)

   FUNCTION
        Compares two times for relative magnitide.  <0 is returned if date1 is
        later than date2, 0 if they are equal, or >0 if date2 is later than
        date1.  NOTE: this is NOT the same ordering as strcmp!

   INPUTS
        date1, date2 - DateStamps to compare

   RESULT
        result -  <0, 0, or >0 based on comparison of two date stamps

   SEE ALSO
        DateStamp(), DateToStr(), StrToDate()
```

dos.library Page 20

```
dos.library/CreateDir                                         dos.library/CreateDir

   NAME
       CreateDir -- Create a new directory

   SYNOPSIS
       lock = CreateDir( name )
       D0                D1

       BPTR CreateDir(char *)

   FUNCTION
       CreateDir creates a new directory with the specified name. An error
       is returned if it fails.  Directories can only be created on
       devices which support them, e.g. disks.  CreateDir returns an
       exclusive lock on the new directory if it succeeds.

   INPUTS
       name - pointer to a null-terminated string

   RESULTS
       lock - BCPL pointer to a lock or NULL for failure.

   SEE ALSO
       Lock(), UnLock()
```

```
dos.library/CreateNewProc                              dos.library/CreateNewProc

   NAME
       CreateNewProc -- Create a new process (V36)

   SYNOPSIS
       process = CreateNewProc(tags)
       D0                      D1

       struct Process *CreateNewProc(struct TagItem *)

       process = CreateNewProcTagList(tags)
       D0                             D1

       struct Process *CreateNewProcTagList(struct TagItem *)

       process = CreateNewProcTags(Tag1, ...)

       struct Process *CreateNewProcTags(ULONG, ...)

   FUNCTION
       This creates a new process according to the tags passed in.  See
       libraries/dostags.h for the tags.

       You must specify one of NP_Seglist or NP_Entry.  NP_Seglist takes a
       seglist (as returned by LoadSeg()).  NP_Entry takes a function
       pointer for the routine to call.

       There are many options, as you can see by examining dos/dostags.h.
       The defaults are for a non-CLI process, with copies of your
       CurrentDir, HomeDir (used for PROGDIR:), priority, consoletask,
       windowptr, and variables.  The input and output filehandles default
       to opens of NIL:, stack to 4000, and others as shown in dostags.h.
       This is a fairly reasonable default setting for creating threads,
       though you may wish to modify it (for example, to give a descriptive
       name to the process.)

       CreateNewProc() is callable from a task, though any actions that
       require doing Dos I/O (DupLock of currentdir, for example) will not
       occur.

   INPUTS
       tags - a pointer to a TagItem array.

   RESULT
       process - The created process, or NULL.

   SEE ALSO
       LoadSeg(), CreateProc(), <dos/dostags.h>
```

```
dos.library/CreateProc                                    dos.library/CreateProc

   NAME

       CreateProc -- Create a new process

   SYNOPSIS
       process = CreateProc( name, pri, seglist, stackSize )
       D0                    D1    D2   D3       D4

       struct MsgPort *CreateProc(char *, LONG, BPTR, LONG)

   FUNCTION
       CreateProc() creates a new AmigaDOS process of name 'name'.  AmigaDOS
       processes are a superset of exec tasks.

       A seglist, as returned by LoadSeg(), is passed as 'seglist'.
       This represents a section of code which is to be run as a new
       process. The code is entered at the first hunk in the segment list,
       which should contain suitable initialization code or a jump to
       such.  A process control structure is allocated from memory and
       initialized.  If you wish to fake a seglist (that will never
       have DOS UnLoadSeg() called on it), use this code:

                   DS.L    0   ;Align to longword
                   DC.L    16  ;Segment "length" (faked)
                   DC.L    0   ;Pointer to next segment
                   ...start of code...

       The size of the root stack upon activation is passed as
       'stackSize'.  'pri' specifies the required priority of the new
       process.  The result will be the process msgport address of the new
       process, or zero if the routine failed.  The argument 'name'
       specifies the new process name.  A zero return code indicates
       error.

       The seglist passed to CreateProc() is not freed when it exits; it
       is up to the parent process to free it, or for the code to unload
       itself.

       Under V36 and later, you probably should use CreateNewProc() instead.

   INPUTS
       name      - pointer to a null-terminated string
       pri       - signed long (range -128 to +127)
       seglist   - BCPL pointer to a seglist
       stackSize - integer (must be a multiple of 4 bytes)

   RESULTS
       process   - pointer to new process msgport

   SEE ALSO
       CreateNewProc(), LoadSeg(), UnLoadSeg()
```

dos.library Page 23

```
dos.library/CurrentDir                                 dos.library/CurrentDir

    NAME
        CurrentDir -- Make a directory lock the current directory

    SYNOPSIS
        oldLock = CurrentDir( lock )
        D0                     D1

        BPTR CurrentDir(BPTR)

    FUNCTION
        CurrentDir() causes a directory associated with a lock to be made
        the current directory.  The old current directory lock is returned.

        A value of zero is a valid result here, this 0 lock represents the
        root of file system that you booted from.

        Any call that has to Open() or Lock() files (etc) requires that
        the current directory be a valid lock or 0.

    INPUTS
        lock - BCPL pointer to a lock

    RESULTS
        oldLock - BCPL pointer to a lock

    SEE ALSO
        Lock(), UnLock(), Open(), DupLock()
```

dos.library Page 24

```
dos.library/DateStamp                                   dos.library/DateStamp

    NAME
        DateStamp -- Obtain the date and time in internal format

    SYNOPSIS
        ds = DateStamp( ds );
        D0              D1

        struct DateStamp *DateStamp(struct DateStamp *)

    FUNCTION
        DateStamp() takes a structure of three longwords that is set to the
        current time.  The first element in the vector is a count of the
        number of days.  The second element is the number of minutes elapsed
        in the day.  The third is the number of ticks elapsed in the current
        minute.  A tick happens 50 times a second.  DateStamp() ensures that
        the day and minute are consistent.  All three elements are zero if
        the date is unset. DateStamp() currently only returns even
        multiples of 50 ticks.  Therefore the time you get is always an even
        number of ticks.

        Time is measured from Jan 1, 1978.

    INPUTS
        ds - pointer a struct DateStamp

    RESULTS
        The array is filled as described and returned (for pre-V36
        compabability).

    SEE ALSO
        DateToStr(), StrToDate(), SetFileDate(), CompareDates()
```

```
dos.library/DateToStr                                  dos.library/DateToStr

   NAME
        DateToStr -- Converts a DateStamp to a string (V36)

   SYNOPSIS
        success = DateToStr( datetime )
        D0                   D1

        BOOL DateToStr(struct DateTime *)

   FUNCTION
        StamptoStr converts an AmigaDOS DateStamp to a human
        readable ASCII string as requested by your settings in the
        DateTime structure.

   INPUTS
        DateTime - a pointer to an initialized DateTime structure.

        The DateTime structure should be initialized as follows:

        dat_Stamp - a copy of the datestamp you wish to convert to
                  ascii.

        dat_Format - a format   byte which specifies the format of the
                  dat_StrDate.  This can be any of the following
                  (note: If value used is something other than those
                  below, the default of FORMAT_DOS is used):

                  FORMAT_DOS:     AmigaDOS format (dd-mmm-yy).

                  FORMAT_INT:     International format (yy-mmm-dd).

                  FORMAT_USA:     American format (mm-dd-yy).

                  FORMAT_CDN:     Canadian format (dd-mm-yy).

                  FORMAT_DEF:     default format for locale.

        dat_Flags - a   flags byte.  The only flag which affects this
                  function is:

                  DTF_SUBST:      If set, a string such as Today,
                                  Monday, etc., will be used instead
                                  of the dat_Format specification if
                                  possible.
                  DTF_FUTURE:     Ignored by this function.

        dat_StrDay - pointer to a buffer to receive the day of the
                  week string.  (Monday, Tuesday, etc.). If null, this
                  string will not be generated.

        dat_StrDate -   pointer to a buffer to receive the date
                  string, in the format requested by dat_Format,
                  subject to possible modifications by DTF_SUBST.  If
                  null, this string will not be generated.

        dat_StrTime -   pointer to a buffer to receive the time of day
                  string. If NULL, this will not be generated.

   RESULT
        success - a zero return indicates that the DateStamp was
                  invalid, and could not be converted.  Non-zero
                  indicates that the call succeeded.

   SEE ALSO
```

```
        DateStamp(), StrtoDate(), <dos/datetime.h>
```

dos.library Page 27

```
dos.library/Delay                                          dos.library/Delay

    NAME
        Delay -- Delay a process for a specified time

    SYNOPSIS
        Delay( ticks )
               D1

        void Delay(ULONG)

    FUNCTION
        The argument 'ticks' specifies how many ticks (50 per second) to
        wait before returning control.

    INPUTS
        ticks - integer

    BUGS
        Due to a bug in the timer.device in V1.2/V1.3, specifying a timeout
        of zero for Delay() can cause the unreliable timer & floppy disk
        operation.  This is fixed in V36 and later.

    SEE ALSO
```

dos.library Page 28

```
dos.library/DeleteFile                                dos.library/DeleteFile

    NAME
        DeleteFile -- Delete a file or directory

    SYNOPSIS
        success = DeleteFile( name )
        D0                    D1

        BOOL DeleteFile(char *)

    FUNCTION
        This attempts to delete the file or directory specified by 'name'.
        An error is returned if the deletion fails. Note that all the files
        within a directory must be deleted before the directory itself can
        be deleted.

    INPUTS
        name - pointer to a null-terminated string

    RESULTS
        success - boolean

    SEE ALSO
```

```
dos.library/DeleteVar                                    dos.library/DeleteVar

   NAME
       DeleteVar -- Deletes a local or environment variable (V36)

   SYNOPSIS
       success = DeleteVar( name, flags )
       D0                   D1    D2

       BOOL DeleteVar(UBYTE *, ULONG )

   FUNCTION
       Deletes a local or environment variable.

   INPUTS
       name    - pointer to an variable name.  Note variable names follow
                 filesystem syntax and semantics.
       flags   - combination of type of var to delete (low 8 bits), and
                 flags to control the behavior of this routine.  Currently
                 defined flags include:

                 GVF_LOCAL_ONLY  - delete a local (to your process) variable.
                 GVF_GLOBAL_ONLY - delete a global environment variable.

                 The default is to delete a local variable if found, otherwise
                 a global environment variable if found (only for LV_VAR).

   RESULT
       success - If non-zero, the variable was sucessfully deleted, FALSE
                  indicates failure.

   BUGS
       LV_VAR is the only type that can be global

   SEE ALSO
       GetVar(), SetVar(), FindVar(), DeleteFile(), <dos/var.h>
```

```
dos.library/DeviceProc                                  dos.library/DeviceProc

   NAME
       DeviceProc -- Return the process MsgPort of specific I/O handler

   SYNOPSIS
       process = DeviceProc( name )
       D0                    D1

       struct MsgPort *DeviceProc (char *)

   FUNCTION
       DeviceProc() returns the process identifier of the process which
       handles the device associated with the specified name. If no
       process handler can be found then the result is zero. If the name
       refers to an assign then a directory lock is returned in IoErr().
       This lock should not be UnLock()ed or Examine()ed (if you wish to do
       so, DupLock() it first).

   BUGS
       In V36, if you try to DeviceProc() something relative to an assign
       made with AssignPath(), it will fail.  This is because there's no
       way to know when to unlock the lock.  If you're writing code for
       V36 or later, it is highly advised you use GetDeviceProc() instead,
       or make your code conditional on V36 to use GetDeviceProc()/
       FreeDeviceProc().

   SEE ALSO
       GetDeviceProc(), FreeDeviceProc(), DupLock(), UnLock(), Examine()
```

dos.library

Page 31

```
dos.library/DoPkt                                           dos.library/DoPkt

    NAME
        DoPkt -- Send a dos packet and wait for reply (V36)

    SYNOPSIS
        result1 = DoPkt(port,action,arg1,arg2,arg3,arg4,arg5)
        D0               D1     D2     D3   D4   D5   D6   D7

        LONG = DoPkt(struct MsgPort *,LONG,LONG,LONG,LONG,LONG,LONG)

    FUNCTION
        Sends a packet to a handler and waits for it to return.  Any secondary
        return will be available in D1 AND from IoErr().  DoPkt() will work
        even if the caller is an exec task and not a process; however it will
        be slower, and may fail for some additional reasons, such as being
        unable to allocate a signal.  DoPkt() uses your pr_MsgPort for the
        reply, and will call pr_PktWait.  (See BUGS regarding tasks, though).

        Only allows 5 arguments to be specified.  For more arguments (packets
        support a maximum of 7) create a packet and use SendPkt()/WaitPkt().

    INPUTS
        port    - pr_MsgPort of the handler process to send to.
        action  - the action requested of the filesystem/handler
        arg1, arg2, arg3, arg4,arg5 - arguments, depend on the action, may not
                   be required.

    RESULT
        result1 - the value returned in dp_Res1, or FALSE if there was some
                  problem in sending the packet or recieving it.

    BUGS
        Using DoPkt() from tasks doesn't work in Dos 2.0. Use
        AllocDosObject(), PutMsg(), and WaitPort()/GetMsg() for a workaround,
        or you can call CreateNewProc() to start a process to do Dos I/O for
        you.

    SEE ALSO
        AllocDosObject(), FreeDosObject(), SendPkt(), WaitPkt(),
        CreateNewProc(), AbortPkt()
```

dos.library

Page 32

```
dos.library/DupLock                                      dos.library/DupLock

    NAME
        DupLock -- Duplicate a lock

    SYNOPSIS
        lock = DupLock( lock )
        D0              D1

        BPTR DupLock(BPTR)

    FUNCTION
        DupLock() is passed a shared filing system lock.  This is the ONLY
        way to obtain a duplicate of a lock... simply copying is not
        allowed.

        Another lock to the same object is then returned.  It is not
        possible to create a copy of a exclusive lock.

        A zero return indicates failure.

    INPUTS
        lock - BCPL pointer to a lock

    RESULTS
        newLock - BCPL pointer to a lock

    SEE ALSO
        Lock(), UnLock(), DupLockFromFH(), ParentOfFH()
```

```
dos.library/DupLockFromFH                              dos.library/DupLockFromFH

   NAME
       DupLockFromFH -- Gets a lock on an open file (V36)

   SYNOPSIS
       lock = DupLockFromFH(fh)
       D0                   D1

       BPTR DupLockFromFH(BPTR)

   FUNCTION
       Obtain a lock on the object associated with fh.  Only works if the
       file was opened using a non-exclusive mode.  Other restrictions may be
       placed on success by the filesystem.

   INPUTS
       fh   - Opened file for which to obtain the lock

   RESULT
       lock - Obtained lock or NULL for failure

   SEE ALSO
       DupLock(), Lock(), UnLock()
```

```
dos.library/EndNotify                                      dos.library/EndNotify

   NAME
       EndNotify -- Ends a notification request (V36)

   SYNOPSIS
       EndNotify(notifystructure)
                       D1

       VOID EndNotify(struct NotifyRequest *)

   FUNCTION
       Removes a notification request.  Safe to call even if StartNotify()
       failed.  For NRF_SEND_MESSAGE, it searches your port for any messages
       about the object in question and removes and replies them before
       returning.

   INPUTS
       notifystructure - a structure passed to StartNotify()

   SEE ALSO
       StartNotify(), <dos/notify.h>
```

```
dos.library/ErrorReport                                   dos.library/ErrorReport

    NAME
        ErrorReport -- Displays a Retry/Cancel requester for an error (V36)

    SYNOPSIS
        status = ErrorReport(code, type, arg1, device)
        D0                   D1    D2    D3    A0

        BOOL ErrorReport(LONG, LONG, ULONG, struct MsgPort *)

    FUNCTION
        Based on the request type, this routine formats the appropriate
        requester to be displayed.  If the code is not understood, it returns
        DOS_TRUE immediately.  Returns DOS_TRUE if the user selects CANCEL or
        if the attempt to put up the requester fails, or if the process
        pr_WindowPtr is -1.  Returns FALSE if the user selects Retry.  The
        routine will retry on DISKINSERTED for appropriate error codes.
        These return values are the opposite of what AutoRequest returns.

        Note: this routine sets IoErr() to code before returning.

    INPUTS
        code   - Error code to put a requester up for.
           Current valid error codes are:
                ERROR_DISK_NOT_VALIDATED
                ERROR_DISK_WRITE_PROTECTED
                ERROR_DISK_FULL
                ERROR_DEVICE_NOT_MOUNTED
                ERROR_NOT_A_DOS_DISK
                ERROR_NO_DISK
                ABORT_DISK_ERROR        /* read/write error */
                ABORT_BUSY              /* you MUST replace... */
        type   - Request type:
                       REPORT_LOCK   - arg1 is a lock (BPTR).
                       REPORT_FH     - arg1 is a filehandle (BPTR).
                        REPORT_VOLUME - arg1 is a volumenode (C pointer).
                        REPORT_INSERT - arg1 is the string for the volumename
                                        (will be split on a ':').
                                        With ERROR_DEVICE_NOT_MOUNTED puts
                                        up the "Please insert..." requester.
        arg1   - variable parameter (see type)
        device - (Optional) Address of handler task for which report is to be
                made.  Only required for REPORT_LOCK, and only if arg1==NULL.

    RESULT
        status - Cancel/Retry indicator (0 means Retry)

    SEE ALSO
        Fault(), IoErr()
```

```
dos.library/ExAll                                             dos.library/ExAll

    NAME
        ExAll -- Examine an entire directory (V36)

    SYNOPSIS
        continue = ExAll(lock, buffer, size, type, control)
        D0               D1    D2      D3    D4    D5

        BOOL ExAll(BPTR,UBYTE *,LONG,LONG,struct ExAllControl *)

    FUNCTION
        Examines an entire directory.

  Lock must be on a directory.  Size is the size of the buffer supplied.
  The buffer will be filled with (partial) ExAllData structures, as
  specified by the type field.

  Type is a value from those shown below that determines which information is
  to be stored in the buffer.  Each higher value adds a new thing to the list
  as described in the table below:-

        ED_NAME         FileName
        ED_TYPE         Type
        ED_SIZE         Size in bytes
        ED_PROTECTION   Protection bits
        ED_DATE         3 longwords of date
        ED_COMMENT      Comment (will be NULL if no comment)

  Thus, ED_NAME gives only filenames, and ED_COMMENT gives everything.

  The ead_Next entry gives a pointer to the next entry in the buffer.  The
  last entry will have NULL in ead_Next.

  The control structure is required so that FFS can keep track if more than
  one call to ExAll is required.  This happens when there are more names in
  a directory than will fit into the buffer.  The format of the control
  structure is as follows:-

  NOTE: the control structure MUST be allocated by AllocDosObject!!!

  Entries:  This field tells the calling application how many entries are
             in the buffer after calling ExAll.  Note: make sure your code
             handles the 0 entries case, including 0 entries with continue
             non-zero.

  LastKey:  This field ABSOLUTELY MUST be initialised to 0 before calling
             ExAll for the first time.  Any other value will cause nasty
             things to happen.  If ExAll returns non-zero, then this field
             should not be touched before making the second and subsequent
             calls to ExAll.  Whenever ExAll returns non-zero, there are more
             calls required before all names have been received.

             As soon as a FALSE return is received then ExAll has completed
             (if IoErr() returns ERROR_NO_MORE_ENTRIES - otherwise it returns
             the error that occured, similar to ExNext.)

  MatchString
             If this field is NULL then all filenames will be returned.  If
             this field is non-null then it is interpreted as a pointer to
             a string that is used to pattern match all file names before
             accepting them and putting them into the buffer.  The default
             AmigaDOS caseless pattern match routine is used.  This string
             MUST have been parsed by ParsePatternNoCase()!

  MatchFunc:
        Contains a pointer to a hook for a routine to decide if the entry
        will be included in the returned list of entries.  The entry is
        filled out first, and then passed to the hook.  If no MatchFunc is
        to be called then this entry should be NULL.  The hook is
        called with the following parameters (as is standard for hooks):

        BOOL = MatchFunc( hookptr, data, typeptr )
                            a0      a1     a2
        (a0 = ptr to hook, a1 = ptr to filled in ExAllData, a2 = ptr
         to longword of type).

        MatchFunc should return FALSE if the entry is not to be
        accepted, otherwise return TRUE.

    Note that Dos will emulate ExAll() using Examine() and ExNext()
    if the handler in question doesn't support the ExAll() packet.

INPUTS
    lock    - Lock on directory to be examined.
    buffer  - Buffer for data returned (MUST be at least word-aligned,
              preferably long-word aligned).
    size    - Size in bytes of 'buffer'.
    type    - Type of data to be returned.
    control - Control data structure (see notes above).  MUST have been
              allocated by AllocDosObject!

RESULT
    continue - Whether or not ExAll is done.  If FALSE is returned, either
               ExAll has completed (IoErr() == ERROR_NO_MORE_ENTRIES), or
               an error occurred (check IoErr()).  If non-zero is returned,
               you MUST call ExAll again until it returns FALSE.

EXAMPLE

eac = AllocDosObject(DOS_EXALLCONTROL,NULL);
if (!eac) ...
...
eac->eac_LastKey = 0;
do {
    more = ExAll(lock, EAData, sizeof(EAData), ED_FOO, eac);
    if ((!more) && (IoErr() != ERROR_NO_MORE_ENTRIES)) {
        /* ExAll failed abnormally */
        break;
    }
    if (eac->eac_Entries == 0) {
        /* ExAll failed normally with no entries */
        continue;                   /* ("more" is *usually* zero) */
    }
    ead = (struct ExAllData *) EAData;
    do {
        /* use ead here */
        ...
        /* get next ead */
        ead = ead->ed_Next;
    } while (ead);

} while (more);
...
FreeDosObject(DOS_EXALLCONTROL,eac);

BUGS
    In V36, there were problems with ExAll (particularily with
    eac_MatchString, and ed_Next with the ramdisk and the emulation
    of it in Dos for handlers that do not support the packet.  It is
    advised you only use this under V37 and later.
```

```
SEE ALSO
    Examine(), ExNext(), ExamineFH(), MatchPatternNoCase(),
    ParsePatternNoCase(), AllocDosObject()
```

```
dos.library/Examine                                        dos.library/Examine

    NAME
        Examine -- Examine a directory or file associated with a lock

    SYNOPSIS
        success = Examine( lock, FileInfoBlock )
        D0                 D1         D2

        BOOL Examine(BPTR,struct FileInfoBlock *)

    FUNCTION
        Examine() fills in information in the FileInfoBlock concerning the
        file or directory associated with the lock. This information
        includes the name, size, creation date and whether it is a file or
        directory.  FileInfoBlock must be longword aligned.  Examine() gives
        a return code of zero if it fails.

        You may make a local copy of the FileInfoBlock, as long as it is
        never passed to ExNext().

    INPUTS
        lock      - BCPL pointer to a lock
        infoBlock - pointer to a FileInfoBlock (MUST be longword aligned)

    RESULTS
        success - boolean

    SPECIAL NOTE
        FileInfoBlock must be longword-aligned.  AllocDosObject() will
        allocate them correctly for you.

    SEE ALSO
        Lock(), UnLock(), ExNext(), ExamineFH(), <dos/dos.h>, AllocDosObject(),
        ExAll()
```

```
dos.library/ExamineFH                                    dos.library/ExamineFH

   NAME
       ExamineFH -- Gets information on an open file (V36)

   SYNOPSIS
       success = ExamineFH(fh, fib)
       D0                  D1  D2

       BOOL ExamineFH(BPTR, struct FileInfoBlock *)

   FUNCTION
       Examines a filehandle and returns information about the file in the
       FileInfoBlock.  There are no guarantees as to whether the fib_Size
       field will reflect any changes made to the file size it was opened,
       though filesystems should attempt to provide up-to-date information
       for it.

   INPUTS
       fh  - Filehandle you wish to examine
       fib - FileInfoBlock, must be longword aligned.

   RESULT
       success - Success/failure indication

   SEE ALSO
       Examine(), ExNext(), ExAll(), Open(), AllocDosObject()
```

```
dos.library/Execute                                     dos.library/Execute

    NAME
        Execute -- Execute a CLI command

    SYNOPSIS
        success = Execute( commandString, input, output )
        D0                 D1             D2     D3

        BOOL Execute(char *, BPTR, BPTR)

    FUNCTION
        This function attempts to execute the string commandString as a
        Shell command and arguments. The string can contain any valid input
        that you could type directly in a Shell, including input and output
        redirection using < and >.  Note that Execute() doesn't return until
        the command(s) in commandstring have returned.

        The input file handle will normally be zero, and in this case
        Execute() will perform whatever was requested in the commandString
        and then return. If the input file handle is nonzero then after the
        (possibly empty) commandString is performed subsequent input is read
        from the specified input file handle until end of that file is
        reached.

        In most cases the output file handle must be provided, and is used
        by the Shell commands as their output stream unless output
        redirection was specified. If the output file handle is set to zero
        then the current window, normally specified as *, is used. Note
        that programs running under the Workbench do not normally have a
        current window.

        Execute() may also be used to create a new interactive Shell process
        just like those created with the NewShell command. In order to do
        this you would call Execute() with an empty commandString, and pass
        a file handle relating to a new window as the input file handle.
        The output file handle would be set to zero. The Shell will read
        commands from the new window, and will use the same window for
        output. This new Shell window can only be terminated by using the
        EndCLI command.

        Under V37, if an input filehandle is passed, and it's either
        interactive or a NIL: filehandle, the pr_ConsoleTask of the new
        process will be set to that filehandle's process (the same applies
        to SystemTagList()).

        For this command to work the program Run must be present in C: in
        versions before V36 (except that in 1.3.2 and any later 1.3 versions,
        the system first checks the resident list for Run).

    INPUTS
        commandString - pointer to a null-terminated string
        input         - BCPL pointer to a file handle
        output        - BCPL pointer to a file handle

    RESULTS
        success - BOOLEAN indicating whether Execute was successful
                  in finding and starting the specified program.  Note this
                  is NOT the return code of the command(s).
    SEE ALSO
        SystemTagList(), NewShell, EndCLI, Run
```

```
dos.library/Exit                                           dos.library/Exit

    NAME
        Exit -- Exit from a program

    SYNOPSIS
        Exit( returnCode )
              D1

        void Exit(LONG)

    FUNCTION
        Exit() is currently for use with programs written as if they
        were BCPL programs.  This function is not normally useful for
        other purposes.

        In general, therefore, please DO NOT CALL THIS FUNCTION!

        In order to exit, C programs should use the C language exit()
        function (note the lower case letter "e").  Assembly programs should
        place a return code in D0, and execute an RTS instruction with
        their original stack ptr.

    IMPLEMENTATION
        The action of Exit() depends on whether the program which called it
        is running as a command under a CLI or not. If the program is
        running under the CLI the command finishes and control reverts to
        the CLI. In this case, returnCode is interpreted as the return code
        from the program.

        If the program is running as a distinct process, Exit() deletes the
        process and release the space associated with the stack, segment
        list and process structure.

    INPUTS
        returnCode - integer

    SEE ALSO
        CreateProc(), CreateNewProc()
```

dos.library Page 43

```
dos.library/ExNext                                          dos.library/ExNext

    NAME
        ExNext -- Examine the next entry in a directory

    SYNOPSIS
        success = ExNext( lock, FileInfoBlock )
        D0                D1        D2

        BOOL ExNext(BPTR, struct FileInfoBlock *)

    FUNCTION
        This routine is passed a directory lock and a FileInfoBlock that
        have been initialized by a previous call to Examine(), or updated
        by a previous call to ExNext().  ExNext() gives a return code of zero
        on failure.  The most common cause of failure is reaching the end
        of the list of files in the owning directory.  In this case, IoErr
        will return ERROR_NO_MORE_ENTRIES and a good exit is appropriate.

        So, follow these steps to examine a directory:
        1) Pass a Lock and a FileInfoBlock to Examine().  The lock must
           be on the directory you wish to examine.
        2) Pass ExNext() the same lock and FileInfoBlock.
        3) Do something with the information returned in the FileInfoBlock.
           Note that the fib_DirEntryType field is positive for directories,
           negative for files.
        4) Keep calling ExNext() until it returns FALSE.  Check IoErr()
           to ensure that the reason for failure was ERROR_NO_MORE_ENTRIES.

        Note: if you wish to recursively scan the file tree and you find
        another directory while ExNext()ing you must Lock that directory and
        Examine() it using a new FileInfoBlock.  Use of the same
        FileInfoBlock to enter a directory would lose important state
        information such that it will be impossible to continue scanning
        the parent directory.  While it is permissible to UnLock() and Lock()
        the parent directory between ExNext() calls, this is NOT recommended.
        Important state information is associated with the parent lock, so
        if it is freed between ExNext() calls this information has to be
        rebuilt on each new ExNext() call, and will significantly slow down
        directory scanning.

        It is NOT legal to Examine() a file, and then to ExNext() from that
        FileInfoBlock.  You may make a local copy of the FileInfoBlock, as
        long as it is never passed back to the operating system.

    INPUTS
        lock - BCPL pointer to a lock originally used for the Examine() call
        infoBlock - pointer to a FileInfoBlock used on the previous Examine()
                    or ExNext() call.

    RESULTS
        success - boolean

    SPECIAL NOTE
        FileInfoBlock must be longword-aligned.  AllocDosObject() will
        allocate them correctly for you.

    SEE ALSO
        Examine(), Lock(), UnLock(), IoErr(), ExamineFH(), AllocDosObject(),
        ExAll()
```

dos.library Page 44

```
dos.library/Fault                                            dos.library/Fault

   NAME
        Fault -- Returns the text associated with a DOS error code (V36)

   SYNOPSIS
        success = Fault(code, header, buffer, len)
        D0               D1     D2      D3     D4

        BOOL Fault(LONG, UBYTE *, UBYTE *, LONG)

   FUNCTION
        This routine obtains the error message text for the given error code.
        The header is prepended to the text of the error message, followed
        by a colon.  Puts a null-terminated string for the error message into
        the buffer.  By convention, error messages should be no longer than 80
        characters (+1 for termination), and preferably no more than 60.
        The value returned by IoErr() is set to the code passed in.  If there
        is no message for the error code, the message will be "Error code
        <number>\n".

   INPUTS
        code   - Error code
        header - header to output before error text
        buffer - Buffer to receive error message.
        len    - Length of the buffer.

   RESULT
        success - Success/failure code.

   SEE ALSO
        IoErr(), SetIoErr(), PrintFault()
```

```
dos.library/FGetC                                          dos.library/FGetC

   NAME
       FGetC -- Read a character from the specified input (buffered) (V36)

   SYNOPSIS
       char = FGetC(fh)
       D0           D1

       LONG FGetC(BPTR)

   FUNCTION
       Reads the next character from the input stream.  A -1 is
       returned when EOF or an error is encountered.  This call is buffered.
       Use Flush() between buffered and unbuffered I/O on a filehandle.

   INPUTS
       fh - filehandle to use for buffered I/O

   RESULT
       char - character read (0-255) or -1

   BUGS
       In V36, after an EOF was read, EOF would always be returned from
       FGetC() from then on.  Starting in V37, it tries to read from the
       handler again each time (unless UnGetC(fh,-1) was called).

   SEE ALSO
       FPutC(), UnGetC(), Flush()
```

```
dos.library/FGets                                          dos.library/FGets

   NAME
       FGets -- Reads a line from the specified input (buffered) (V36)

   SYNOPSIS
       buffer = FGets(fh, buf, len)
       D0             D1  D2   D3

       UBYTE *FGets(BPTR, UBYTE *, ULONG)

   FUNCTION
       This routine reads in a single line from the specified input stopping
       at a NEWLINE character or EOF.  In either event, UP TO the number of
       len specified bytes minus 1 will be copied into the buffer.  Hence if
       a length of 50 is passed and the input line is longer than 49 bytes,
       it will return 49 characters.  It returns the buffer pointer normally,
       or NULL if EOF is the first thing read.

       If terminated by a newline, the newline WILL be the last character in
       the buffer.  This is a buffered read routine.  The string read in IS
       null-terminated.

   INPUTS
       fh  - filehandle to use for buffered I/O
       buf - Area to read bytes into.
       len - Number of bytes to read, must be > 0.

   RESULT
       buffer - Pointer to buffer passed in, or NULL for immediate EOF or for
                an error.  If NULL is returnd for an EOF, IoErr() will return
                0.

   SEE ALSO
       FRead(), FGets(), FPuts()
```

dos.library

Page 47

```
dos.library/FilePart                                     dos.library/FilePart

   NAME
       FilePart -- Returns the last component of a path (V36)

   SYNOPSIS
       fileptr = FilePart( path )
       D0                  D1

       UBYTE *FilePart( UBYTE * )

   FUNCTION
       This function returns a pointer to the last component of a string path
       specification, which will normally be the file name.  If there is only
       one component, it returns a pointer to the beginning of the string.

   INPUTS
       path - pointer to an path string.  May be relative to the current
              directory or the current disk.

   RESULT
       fileptr - pointer to the last component of the path.

   EXAMPLE
       FilePart("xxx:yyy/zzz/qqq") would return a pointer to the first 'q'.
       FilePart("xxx:yyy") would return a pointer to the first 'y').

   SEE ALSO
       PathPart(), AddPart()
```

dos.library

Page 48

```
dos.library/FindArg                                       dos.library/FindArg

   NAME
       FindArg - find a keyword in a template (V36)

   SYNOPSIS
       index = FindArg(template, keyword)
       D0               D1        D2

       LONG FindArg(UBYTE *, UBYTE *)

   FUNCTION
       Returns the argument number of the keyword, or -1 if it is not a
       keyword for the template.  Abbreviations are handled.

   INPUTS
       keyword  - keyword to search for in template
       template - template string to search

   RESULT
       index - number of entry in template, or -1 if not found

   BUGS
       In earlier published versions of the autodoc, keyword and template
       were backwards.

   SEE ALSO
       ReadArgs(), ReadItem(), FreeArgs()
```

```
dos.library/FindCliProc                              dos.library/FindCliProc

   NAME
       FindCliProc -- returns a pointer to the requested CLI process (V36)

   SYNOPSIS
       proc = FindCliProc(num)
       D0                 D1

       struct Process *FindCliProc(LONG)

   FUNCTION
       This routine returns a pointer to the CLI process associated with the
       given CLI number.  If the process isn't an active CLI process, NULL is
       returned.  NOTE: should normally be called inside a Forbid(), if you
       must use this function at all.

   INPUTS
       num  - Task number of CLI process

   RESULT
       proc - Pointer to given CLI process

   SEE ALSO
       Cli(), Forbid(), MaxCli()
```

```
dos.library/FindDosEntry                            dos.library/FindDosEntry

   NAME
       FindDosEntry -- Finds a specific Dos List entry (V36)

   SYNOPSIS
       newdlist = FindDosEntry(dlist,name,flags)
       D0                      D1    D2   D3

       struct DosList *FindDosEntry(struct DosList *,UBYTE *,ULONG)

   FUNCTION
       Locates an entry on the device list.  Starts with the entry dlist.
       NOTE: must be called with the device list locked, no references may be
       made to dlist after unlocking.

   INPUTS
       dlist    - The device entry to start with.
       name     - Name of device entry (without ':') to locate.
       flags    - Search control flags.  Use the flags you passed to
                  LockDosList, or a subset of them.  LDF_READ/LDF_WRITE are
                  not required for this call.

   RESULT
       newdlist - The device entry or NULL

   SEE ALSO
       AddDosEntry(), RemDosEntry(), NextDosEntry(), LockDosList(),
       MakeDosEntry(), FreeDosEntry()
```

dos.library Page 51

```
dos.library/FindSegment                              dos.library/FindSegment

   NAME
        FindSegment - Finds a segment on the resident list (V36)

   SYNOPSIS
        segment = FindSegment(name, start, system)
        D0                    D1     D2     D3

        struct Segment *FindSegment(char *, struct Segment *, LONG)

   FUNCTION
        Finds a segment on the Dos resident list by name and type, starting
        at the segment AFTER 'start', or at the beginning if start is NULL.
        If system is zero, it will only return nodes with a seg_UC of 0
        or more.  It does NOT increment the seg_UC, and it does NOT do any
        locking of the list.  You must Forbid() lock the list to use this
        call.

        To use an entry you have found, you must: if the seg_UC is 0 or more,
        increment it, and decrement it (under Forbid()!) when you're done
        the the seglist.

        The other values for seg_UC are:
                -1   - system module, such as a filesystem or shell
                -2   - resident shell command
                -999 - disabled internal command, ignore
        Negative values should never be modified.  All other negative
        values between 0 and -32767 are reserved to AmigaDos and should not
        be used.

   INPUTS
        name   - name of segment to find
        start  - segment to start the search after
        system - true for system segment, false for normal segments

   RESULT
        segment - the segment found or NULL

   SEE ALSO
        AddSegment(), RemSegment(), Forbid()
```

dos.library Page 52

```
dos.library/FindVar                                      dos.library/FindVar

   NAME
        FindVar -- Finds a local variable (V36)

   SYNOPSIS
        var = FindVar( name, type )
        D0             D1    D2

        struct LocalVar * FindVar(UBYTE *, ULONG )

   FUNCTION
        Finds a local variable structure.

   INPUTS
        name - pointer to an variable name.  Note variable names follow
               filesystem syntax and semantics.

        type - type of variable to be found (see <dos/var.h>)

   RESULT

        var  - pointer to a LocalVar structure or NULL

   SEE ALSO
        GetVar(), SetVar(), DeleteVar(), <dos/var.h>
```

```
dos.library/Flush                                           dos.library/Flush

   NAME
        Flush -- Flushes buffers for a buffered filehandle (V36)

   SYNOPSIS
        success = Flush(fh)
        D0              D1

        BOOL Flush(BPTR)

   FUNCTION
        Flushes any pending buffered writes to the filehandle.  All buffered
        writes will also be flushed on Close().  If the filehandle was being
        used for input, it drops the buffer, and tries to Seek() back to the
        last read position  (so subsequent reads or writes will occur at the
        expected position in the file).

   INPUTS
        fh      - Filehandle to flush.

   RESULT
        success - Success or failure.

   SEE ALSO
        FputC(), FGetC(), UnGetC(), Seek(), Close()
```

```
dos.library/Format                                         dos.library/Format

   NAME
        Format -- Causes a filesystem to initialize itself (V36)

   SYNOPSIS
        success = Format(filesystem, volumename, dostype)
        D0               D1          D2          D3

        BOOL Format(UBYTE *, UBYTE *, ULONG)

   FUNCTION
        Interface for initializing new media on a device.  This causes the
        filesystem to write out an empty disk structure to the media, which
        should then be ready for use.  This assumes the media has been low-
        level formatted and verified already.

   INPUTS
        filesystem - Name of device to be formatted.  ':' must be supplied.
        volumename - Name for volume (if supported).  No ':'.
        dostype    - Type of format, if filesystem supports multiple types.

   RESULT
        success - Success/failure indicator.

   BUGS
        Existed, but was non-functional in V36 dos.  (The volumename wasn't
        converted to a BSTR.)  Workaround: require V37, or under V36
        convert volumename to a BPTR to a BSTR before calling Format().
        Note: a number of printed packet docs for ACTION_FORMAT are wrong
        as to the arguments.

   SEE ALSO
```

dos.library

Page 55

```
dos.library/FPutC                                         dos.library/FPutC

   NAME
       FPutC -- Write a character to the specified output (buffered) (V36)

   SYNOPSIS
       char = FPutC(fh, char)
       D0           D1   D2

       LONG FPutC(BPTR, UBYTE)

   FUNCTION
       Writes a single character to the output stream.  This call is
       buffered.  Use Flush() between buffered and unbuffered I/O on a
       filehandle.  Interactive filehandles are flushed automatically
       on a newline, return, '\0', or line feed.

   INPUTS
       fh   - filehandle to use for buffered I/O
       char - character to write

   RESULT
       char - either the character written, or EOF for an error.

   SEE ALSO
       FGetC(), UnGetC(), Flush()
```

dos.library

Page 56

```
dos.library/FPuts                                         dos.library/FPuts

   NAME
       FPuts -- Writes a string the the specified output (buffered) (V36)

   SYNOPSIS
       error = FPuts(fh, str)
       D0            D1  D2

       LONG FPuts(BPTR, UBYTE *)

   FUNCTION
       This routine writes an unformatted string to the filehandle.  No
       newline is appended to the string and the length actually written is
       returned.  This routine is buffered.

   INPUTS
       fh    - filehandle to use for buffered I/O
       str   - Null-terminated string to be written to default output

   RESULT
       error - 0 normally, otherwise -1.  Note that this is opposite of
               most other Dos functions, which return success.

   SEE ALSO
       FGets(), FPutC(), FWrite(), PutStr()
```

```
dos.library/FRead                                             dos.library/FRead

   NAME
        FRead -- Reads a number of blocks from an input (buffered) (V36)

   SYNOPSIS
        count = FRead(fh, buf, blocklen, blocks)
        D0             D1  D2     D3        D4

        LONG FRead(BPTR, UBYTE *, ULONG, ULONG)

   FUNCTION
        Attempts to read a number of blocks, each blocklen long, into the
        specified buffer from the input stream.  May return less than
        the number of blocks requested, either due to EOF or read errors.
        This call is buffered.

   INPUTS
        fh       - filehandle to use for buffered I/O
        buf      - Area to read bytes into.
        blocklen - number of bytes per block.  Must be > 0.
        blocks   - number of blocks to read.  Must be > 0.

   RESULT
        count - Number of _blocks_ read, or 0 for EOF.  On an error, the
                number of blocks actually read is returned.

   BUGS
        Doesn't clear IoErr() before starting.  If you want to find out
        about errors, use SetIoErr(0L) before calling.

   SEE ALSO
        FGetC(), FWrite(), FGets()
```

```
dos.library/FreeArgs                                       dos.library/FreeArgs

   NAME
        FreeArgs - Free allocated memory after ReadArgs() (V36)

   SYNOPSIS
        FreeArgs(rdargs)
                 D1

        void FreeArgs(struct RDArgs *)

   FUNCTION
        Frees memory allocated to return arguments in from ReadArgs().  If
        ReadArgs allocated the RDArgs structure it will be freed.

   INPUTS
        rdargs - structure returned from ReadArgs()

   SEE ALSO
        ReadArgs(), ReadItem(), FindArg()
```

dos.library

```
dos.library/FreeDeviceProc                          dos.library/FreeDeviceProc

   NAME
       FreeDeviceProc -- Releases port returned by GetDeviceProc() (V36)

   SYNOPSIS
       FreeDeviceProc(devproc)
                       D1

       void FreeDeviceProc(struct DevProc *)

   FUNCTION
       Frees up the structure created by GetDeviceProc(), and any associated
       temporary locks.

       Decrements the counter incremented by GetDeviceProc().  The counter
       is in an extension to the 1.3 process structure.  After calling
       FreeDeviceProc(), do not use the port or lock again!  It is safe to
       call FreeDeviceProc(NULL).

   INPUTS
       devproc - A value returned by GetDeviceProc()

   BUGS
       Counter not currently active in 2.0.

   SEE ALSO
       GetDeviceProc(), DeviceProc(), AssignLock(), AssignLate(),
       AssignPath()
```

dos.library

```
dos.library/FreeDosEntry                                  dos.library/FreeDosEntry

   NAME
       FreeDosEntry -- Frees an entry created by MakeDosEntry (V36)

   SYNOPSIS
       FreeDosEntry(dlist)
                     D1

       void FreeDosEntry(struct DosList *)

   FUNCTION
       Frees an entry created by MakeDosEntry().  This routine should be
       eliminated and replaced by a value passed to FreeDosObject()!

   INPUTS
       dlist - DosList to free.

   SEE ALSO
       AddDosEntry(), RemDosEntry(), FindDosEntry(), LockDosList(),
       NextDosEntry(), MakeDosEntry()
```

```
dos.library/FreeDosObject                          dos.library/FreeDosObject

   NAME
       FreeDosObject -- Frees an object allocated by AllocDosObject() (V36)

   SYNOPSIS
       FreeDosObject(type, ptr)
                     D1    D2

       void FreeDosObject(ULONG, void *)

   FUNCTION
       Frees an object allocated by AllocDosObject().  Do NOT call for
       objects allocated in any other way.

   INPUTS
       type - type passed to AllocDosObject()
       ptr  - ptr returned by AllocDosObject()

   SEE ALSO
       AllocDosObject(), <dos/dos.h>
```

```
dos.library/FWrite                                        dos.library/FWrite

   NAME
       FWrite -- Writes a number of blocks to an output (buffered) (V36)

   SYNOPSIS
       count = FWrite(fh, buf, blocklen, blocks)
       D0             D1  D2   D3        D4

       LONG FWrite(BPTR, UBYTE *, ULONG, ULONG)

   FUNCTION
       Attempts to write a number of blocks, each blocklen long, from the
       specified buffer to the output stream.  May return less than the
       number of blocks requested, if there is some error such as a full
       disk or r/w error.  This call is buffered.

   INPUTS
       fh       - filehandle to use for buffered I/O
       buf      - Area to write bytes from.
       blocklen - number of bytes per block.  Must be > 0.
       blocks   - number of blocks to read.  Must be > 0.

   RESULT
       count - Number of _blocks_ written.  On an error, the number of
               blocks actually written is returned.

   BUGS
       Doesn't clear IoErr() before starting.  If you want to find out
       about errors, use SetIoErr(0L) before calling.

   SEE ALSO
       FPutC(), FRead(), FPuts()
```

dos.library

Page 63

```
dos.library/GetArgStr                                   dos.library/GetArgStr

   NAME
       GetArgStr -- Returns the arguments for the process (V36)

   SYNOPSIS
       ptr = GetArgStr()
       D0

       UBYTE *GetArgStr(void)

   FUNCTION
       Returns a pointer to the (null-terminated) arguments for the program
       (process).  This is the same string passed in a0 on startup from CLI.

   RESULT
       ptr - pointer to arguments

   SEE ALSO
       SetArgStr(), RunCommand()
```

dos.library

Page 64

```
dos.library/GetConsoleTask                         dos.library/GetConsoleTask

   NAME
       GetConsoleTask -- Returns the default console for the process (V36)

   SYNOPSIS
       port = GetConsoleTask()
       D0

       struct MsgPort *GetConsoleTask(void)

   FUNCTION
       Returns the default console task's port (pr_ConsoleTask) for the
       current process.

   RESULT
       port - The pr_MsgPort of the console handler, or NULL.

   SEE ALSO
       SetConsoleTask(), Open()
```

```
dos.library/GetCurrentDirName                    dos.library/GetCurrentDirName

   NAME
        GetCurrentDirName -- returns the current directory name (V36)

   SYNOPSIS
        success = GetCurrentDirName(buf, len)
        D0                          D1   D2

        BOOL GetCurrentDirName(char *, LONG)

   FUNCTION
        Extracts the current directory name from the CLI structure and puts it
        into the buffer.  If the buffer is too small, the name is truncated
        appropriately and a failure code returned.  If no CLI structure is
        present, a null string is returned in the buffer, and failure from
        the call (with IoErr() == ERROR_OBJECT_WRONG_TYPE);

   INPUTS
        buf     - Buffer to hold extracted name
        len     - Number of bytes of space in buffer

   RESULT
        success - Success/failure indicator

   BUGS
        In V36, this routine didn't handle 0-length buffers correctly.

   SEE ALSO
        SetCurrentDirName()
```

```
dos.library/GetDeviceProc                              dos.library/GetDeviceProc

   NAME
        GetDeviceProc -- Finds a handler to send a message to (V36)

   SYNOPSIS
        devproc = GetDeviceProc(name, devproc)
          D0                     D1     D2

        struct DevProc *GetDeviceProc(UBYTE *, struct DevProc *)

   FUNCTION
        Finds the handler/filesystem to send packets regarding 'name' to.
        This may involve getting temporary locks.  It returns a structure
        that includes a lock and msgport to send to to attempt your operation.
        It also includes information on how to handle multiple-directory
        assigns (by passing the DevProc back to GetDeviceProc() until it
        returns NULL).

        The initial call to GetDeviceProc() should pass NULL for devproc.  If
        after using the returned DevProc, you get an ERROR_OBJECT_NOT_FOUND,
        and (devproc->dvp_Flags & DVPF_ASSIGN) is true, you should call
        GetDeviceProc() again, passing it the devproc structure.  It will
        either return a modified devproc structure, or NULL (with
        ERROR_NO_MORE_ENTRIES in IoErr()).  Continue until it returns NULL.

        This call also increments the counter that locks a handler/fs into
        memory.  After calling FreeDeviceProc(), do not use the port or lock
        again!

   INPUTS
        name    - name of the object you wish to access.  This can be a
                  relative path ("foo/bar"), relative to the current volume
                  (":foo/bar"), or relative to a device/volume/assign
                  ("foo:bar").
        devproc - A value returned by GetDeviceProc() before, or NULL

   RESULT
        devproc - a pointer to a DevProc structure or NULL

   BUGS
        Counter not currently active in 2.0.
        In 2.0 and 2.01, you HAD to check DVPF_ASSIGN before calling it again.
        This was fixed for the 2.02 release of V36.

   SEE ALSO
        FreeDeviceProc(), DeviceProc(), AssignLock(), AssignLate(),
        AssignPath()
```

dos.library Page 67

```
dos.library/GetFileSysTask                              dos.library/GetFileSysTask

   NAME
       GetFileSysTask -- Returns the default filesystem for the process (V36)

   SYNOPSIS
       port = GetFileSysTask()
       D0

       struct MsgPort *GetFileSysTask(void)

   FUNCTION
       Returns the default filesystem task's port (pr_FileSystemTask) for the
       current process.

   RESULT
       port - The pr_MsgPort of the filesystem, or NULL.

   SEE ALSO
       SetFileSysTask(), Open()
```

dos.library Page 68

```
dos.library/GetProgramDir                                  dos.library/GetProgramDir

   NAME
       GetProgramDir -- Returns a lock on the directory the program was loaded
                        from (V36)

   SYNOPSIS
       lock = GetProgramDir()
       D0

       BPTR GetProgramDir(void)

   FUNCTION
       Returns a shared lock on the directory the program was loaded from.
       This can be used for a program to find data files, etc, that are stored
       with the program, or to find the program file itself.  NULL returns are
       valid, and may occur, for example, when running a program from the
       resident list.  You should NOT unlock the lock.

   RESULT
       lock - A lock on the directory the current program was loaded from,
              or NULL if loaded from resident list, etc.

   BUGS
       Should return a lock for things loaded via resident.  Perhaps should
       return currentdir if NULL.

   SEE ALSO
       SetProgramDir(), Open()
```

dos.library Page 69

```
dos.library/GetProgramName                         dos.library/GetProgramName

   NAME
       GetProgramName -- Returns the current program name (V36)

   SYNOPSIS
       success = GetProgramName(buf, len)
       D0                       D1   D2

       BOOL GetProgramName(char *, LONG)

   FUNCTION
       Extracts the program name from the CLI structure and puts it
       into the buffer.  If the buffer is too small, the name is truncated
       present, a null string is returned in the buffer, and failure from
       the call (with IoErr() == ERROR_OBJECT_WRONG_TYPE);

   INPUTS
       buf     - Buffer to hold extracted name
       len     - Number of bytes of space in buffer

   RESULT
       success - Success/failure indicator

   SEE ALSO
       SetProgramName()
```

dos.library Page 70

```
dos.library/GetPrompt                                    dos.library/GetPrompt

   NAME
       GetPrompt -- Returns the prompt for the current process (V36)

   SYNOPSIS
       success = GetPrompt(buf, len)
       D0                  D1   D2

       BOOL GetPrompt(char *, LONG)

   FUNCTION
       Extracts the prompt string from the CLI structure and puts it
       into the buffer.  If the buffer is too small, the string is truncated
       appropriately and a failure code returned.  If no CLI structure is
       present, a null string is returned in the buffer, and failure from
       the call (with IoErr() == ERROR_OBJECT_WRONG_TYPE);

   INPUTS
       buf     - Buffer to hold extracted prompt
       len     - Number of bytes of space in buffer

   RESULT
       success - Success/failure indicator

   SEE ALSO
       SetPrompt()
```

dos.library Page 71

```
dos.library/GetVar                                     dos.library/GetVar

   NAME
       GetVar -- Returns the value of a local or global variable (V36)

   SYNOPSIS
       len = GetVar( name, buffer, size, flags )
       D0              D1      D2      D3     D4

       LONG GetVar( UBYTE *, UBYTE *, LONG, ULONG )

   FUNCTION
       Gets the value of a local or environment variable.  It is advised to
       only use ASCII strings inside variables, but not required.  This stops
       putting characters into the destination when a \n is hit, unless
       GVF_BINARY_VAR is specified.  (The \n is not stored in the buffer.)

   INPUTS
       name   - pointer to a variable name.
       buffer - a user allocated area which will be used to store
                the value associated with the variable.
       size   - length of the buffer region in bytes.
       flags  - combination of type of var to get value of (low 8 bits), and
                flags to control the behavior of this routine.  Currently
                defined flags include:

                       GVF_GLOBAL_ONLY - tries to get a global env variable.
                       GVF_LOCAL_ONLY  - tries to get a local variable.
                       GVF_BINARY_VAR  - don't stop at \n

                The default is to try to get a local variable first, then
                to try to get a global environment variable.

   RESULT
       len -   Size of environment variable.  -1 indicates that the
               variable was not defined (if IoErr() returns
               ERROR_OBJECT_NOT_FOUND - it returns ERROR_BAD_NUMBER if
               you specify a size of 0).  If the value would overflow
               the user buffer, the buffer is truncated.  The buffer
               returned is null-terminated (even if GVF_BINARY_VAR is
               used).  The number of characters put in the buffer (not
               including '\0') is returned, and IoErr() will return the
               the size of the variable.

   BUGS
       LV_VAR is the only type that can be global.
       Under V36, we documented (and it returned) the size of the variable,
       not the number of characters transferred.  For V37 this was changed
       to the number of characters put in the buffer, and the total size
       of the variable is put in IoErr().

   SEE ALSO
       SetVar(), DeleteVar(), FindVar(), <dos/var.h>
```

dos.library Page 72

```
dos.library/Info                                                 dos.library/Info

   NAME
       Info -- Returns information about the disk

   SYNOPSIS
       success = Info( lock, parameterBlock )
       D0                D1      D2

       BOOL Info(BPTR, struct InfoData *)

   FUNCTION
       Info() can be used to find information about any disk in use.
       'lock' refers to the disk, or any file on the disk. The parameter
       block is returned with information about the size of the disk,
       number of free blocks and any soft errors.

   INPUTS
       lock           - BCPL pointer to a lock
       parameterBlock - pointer to an InfoData structure
                        (longword aligned)

   RESULTS
       success - boolean

   SPECIAL NOTE:
       Note that InfoData structure must be longword aligned.
```

```
dos.library/Inhibit                                     dos.library/Inhibit

   NAME
        Inhibit -- Inhibits access to a filesystem (V36)

   SYNOPSIS
        success = Inhibit(filesystem, flag)
        D0                    D1        D2

        BOOL Inhibit(char *,LONG)

   FUNCTION
        Sends an ACTION_INHIBIT packet to the indicated handler.  This stops
        all activity by the handler until uninhibited.  When uninhibited,
        anything may have happened to the disk in the drive, or there may no
        longer be one.

   INPUTS
        filesystem - Name of device to inhibit (with ':')
        flag       - New status.  DOSTRUE = inhibited, FALSE = uninhibited

   RESULT
        success    - Success/failure indicator

   SEE ALSO
```

```
dos.library/Input                                         dos.library/Input

    NAME
        Input -- Identify the program's initial input file handle

    SYNOPSIS
        file = Input()
        D0

        BPTR Input(void)

    FUNCTION
        Input() is used to identify the initial input stream allocated when
        the program was initiated.  Never close the filehandle returned by
        Input!

    RESULTS
        file - BCPL pointer to a file handle

    SEE ALSO
        Output(), SelectInput()
```

```
dos.library/InternalLoadSeg                         dos.library/InternalLoadSeg

   NAME
       InternalLoadSeg -- Low-level load routine (V36)

   SYNOPSIS
       seglist = InternalLoadSeg(fh,table,functionarray,stack)
       D0                        D0  A0         A1        A2

       BPTR InternalLoadSeg(BPTR,BPTR,LONG *,LONG *)

   FUNCTION
       Loads from fh.  Table is used when loading an overlay, otherwise
       should be NULL.  Functionarray is a pointer to an array of functions.
       Note that the current Seek position after loading may be at any point
       after the last hunk loaded.  The filehandle will not be closed.  If a
       stacksize is encoded in the file, the size will be stuffed in the
       LONG pointed to by stack.  This LONG should be initialized to your
       default value: InternalLoadSeg() will not change it if no stacksize
       is found. Clears unused portions of Code and Data hunks (as well as
       BSS hunks).  (This also applies to LoadSeg() and NewLoadSeg()).

       If the file being loaded is an overlaid file, this will return
       -(seglist).  All other results will be positive.

       NOTE to overlay users: InternalLoadSeg() does NOT return seglist in
       both D0 and D1, as LoadSeg does.  The current ovs.asm uses LoadSeg(),
       and assumes returns are in D1.  We will support this for LoadSeg()
       ONLY.

   INPUTS
       fh            - Filehandle to load from.
       table         - When loading an overlay, otherwise ignored.
       functionarray - Array of function to be used for read, alloc, and free.
          FuncTable[0] ->  Actual = ReadFunc(readhandle,buffer,length),DOSBase
                           D0                D1         A0     D0      A6
          FuncTable[1] ->  Memory = AllocFunc(size,flags), Execbase
                           D0                 D0   D1      a6
          FuncTable[2] ->  FreeFunc(memory,size), Execbase
                                    A1     D0     a6
       stack         - Pointer to storage (ULONG) for stacksize.

   RESULT
       seglist       - Seglist loaded or NULL.  NOT returned in D1!

   BUGS
       Really should use tags.

   SEE ALSO
       LoadSeg(), UnLoadSeg(), NewLoadSeg(), InternalUnLoadSeg()
```

```
dos.library/InternalUnLoadSeg                     dos.library/InternalUnLoadSeg

   NAME
       InternalUnLoadSeg -- Unloads a seglist loaded with InternalLoadSeg() (V36)

   SYNOPSIS
       success = InternalUnLoadSeg(seglist,FreeFunc)
         D0                          D1       A1

       BOOL InternalUnLoadSeg(BPTR,void (*)(char *,ULONG))

   FUNCTION
       Unloads a seglist using freefunc to free segments.  Freefunc is called
       as for InternalLoadSeg.  NOTE: will call Close() for overlaid
       seglists.

   INPUTS
       seglist  - Seglist to be unloaded
       FreeFunc - Function called to free memory

   RESULT
       success - returns whether everything went OK (since this may close
                 files).  Also returns FALSE if seglist was NULL.

   BUGS
       Really should use tags

   SEE ALSO
       LoadSeg(), UnLoadSeg(), InternalLoadSeg(), NewUnLoadSeg(), Close()
```

```
dos.library/IoErr                                          dos.library/IoErr

    NAME
        IoErr -- Return extra information from the system

    SYNOPSIS
        error = IoErr()
          D0

        LONG IoErr(void)

    FUNCTION
        Most I/O routines return zero to indicate an error. When this
        happens (or whatever the defined error return for the routine)
        this routine may be called to determine more information. It is
        also used in some routines to pass back a secondary result.

        Note: there is no guarantee as to the value returned from IoErr()
        after a successful operation, unless to specified by the routine.

    RESULTS
        error - integer

    SEE ALSO
        Fault(), PrintFault(), SetIoErr()
```

```
dos.library/IsFileSystem                              dos.library/IsFileSystem

   NAME
        IsFileSystem -- returns whether a Dos handler is a filesystem (V36)

   SYNOPSIS
        result = IsFileSystem(name)
        D0                     D1

        BOOL IsFileSystem(char *)

   FUNCTION
        Returns whether the device is a filesystem or not.  A filesystem
        supports seperate files storing information.  It may also support
        sub-directories, but is not required to.  If the filesystem doesn't
        support this new packet, IsFileSystem() will use Lock(":",...) as
        an indicator.

   INPUTS
        name   - Name of device in question, with trailing ':'.

   RESULT
        result - Flag to indicate if device is a file system

   SEE ALSO
        Lock()
```

```
dos.library/IsInteractive                          dos.library/IsInteractive

    NAME
        IsInteractive -- Discover whether a file is "interactive"

    SYNOPSIS
        status = IsInteractive( file )
        D0                      D1

        BOOL IsInteractive(BPTR)

    FUNCTION
        The return value 'status' indicates whether the file associated
        with the file handle 'file' is connected to a virtual terminal.

    INPUTS
        file - BCPL pointer to a file handle

    RESULTS
        status - boolean

    SEE ALSO
```

```
dos.library/LoadSeg                                          dos.library/LoadSeg

    NAME
        LoadSeg -- Scatterload a loadable file into memory

    SYNOPSIS
        seglist = LoadSeg( name )
        D0                 D1

        BPTR LoadSeg(char *)

    FUNCTION
        The file 'name' should be a load module produced by the linker.
        LoadSeg() scatterloads the CODE, DATA and BSS segments into memory,
        chaining together the segments with BPTR's on their first words.
        The end of the chain is indicated by a zero.  There can be any number
        of segments in a file.  All necessary relocation is handled by
        LoadSeg().

        In the event of an error any blocks loaded will be unloaded and a
        NULL result returned.

        If the module is correctly loaded then the output will be a pointer
        at the beginning of the list of blocks. Loaded code is unloaded via
        a call to UnLoadSeg().

    INPUTS
        name - pointer to a null-terminated string

    RESULTS
        seglist - BCPL pointer to a seglist

    SEE ALSO
        UnLoadSeg(), InternalLoadSeg(), InternalUnLoadSeg(), CreateProc(),
        CreateNewProc(), NewLoadSeg().
```

```
dos.library/Lock                                              dos.library/Lock

    NAME
        Lock -- Lock a directory or file

    SYNOPSIS
        lock  = Lock( name, accessMode )
        D0            D1        D2

        BPTR Lock(char *, LONG)

    FUNCTION
        A filing system lock on the file or directory 'name' is returned if
        possible.

        If the accessMode is ACCESS_READ, the lock is a shared read lock;
        if the accessMode is ACCESS_WRITE then it is an exclusive write
        lock.  Do not use random values for mode.

        If Lock() fails (that is, if it cannot obtain a filing system lock
        on the file or directory) it returns a zero.

        Tricky assumptions about the internal format of a lock are unwise,
        as are any attempts to use the fl_Link or fl_Access fields.

    INPUTS
        name       - pointer to a null-terminated string
        accessMode - integer

    RESULTS
        lock - BCPL pointer to a lock

    SEE ALSO
        UnLock(), DupLock(), ChangeMode(), NameFromLock(), DupLockFromFH()
```

```
dos.library/LockDosList                                dos.library/LockDosList

    NAME
        LockDosList -- Locks the specified Dos Lists for use (V36)

    SYNOPSIS
        dlist = LockDosList(flags)
        D0                  D1

        struct DosList *LockDosList(ULONG)

    FUNCTION
        Locks the dos device list in preparation to walk the list.
        If the list is 'busy' then this routine will not return until it is
        available.  This routine "nests": you can call it multiple times, and
        then must unlock it the same number of times.  The dlist
        returned is NOT a valid entry: it is a special value.  Note that
        for 1.3 compatibility, it also does a Forbid() - this will probably
        be removed at some future time.  The 1.3 Forbid() locking of this
        list had some race conditions.  The pointer returned by this is NOT
        an actual DosList pointer - you should use on of the other DosEntry
        calls to get actual pointers to DosList structures (such as
        NextDosEntry()), passing the value returned by LockDosList()
        as the dlist value.

    INPUTS
        flags - Flags stating which types of nodes you want to lock.

    RESULT
        dlist - Pointer to the head of the list.  NOT a valid node!

    SEE ALSO
        AttemptLockDosList(), UnLockDosList(), Forbid(), NextDosEntry()
```

```
dos.library/LockRecord                                 dos.library/LockRecord

   NAME
        LockRecord -- Locks a portion of a file (V36)

   SYNOPSIS
        success = LockRecord(fh,offset,length,mode,timeout)
        D0                   D1    D2     D3    D4    D5

        ULONG LockRecord(BPTR,ULONG,ULONG,ULONG,ULONG)

   FUNCTION
        This locks a portion of a file for exclusive access.  Timeout is how
        long to wait in ticks (1/50 sec) for the record to be available.

        Valid modes are:
                REC_EXCLUSIVE
                REC_EXCLUSIVE_IMMED
                REC_SHARED
                REC_SHARED_IMMED
        For the IMMED modes, the timeout is ignored.

        Record locks are tied to the filehandle used to create them.  The
        same filehandle can get any number of exclusive locks on the same
        record, for example.  These are cooperative locks, they only
        affect other people calling LockRecord().

   INPUTS
        fh      - File handle for which to lock the record
        offset  - Record start position
        length  - Length of record in bytes
        mode    - Type of lock requester
        timeout - Timeout interval in ticks.  0 is legal.

   RESULT
        success - Success or failure

   BUGS
        In 2.0 through 2.02 (V36), LockRecord() only worked in the ramdisk.
        Attempting to lock records on the disk filesystem causes a crash.
        This was fixed for V37.

   SEE ALSO
        LockRecords(), UnLockRecord(), UnLockRecords()
```

```
dos.library/LockRecords                               dos.library/LockRecords

   NAME
        LockRecords -- Lock a series of records (V36)

   SYNOPSIS
        success = LockRecords(record_array,timeout)
        D0                    D1           D2

        BOOL LockRecords(struct RecordLock *,ULONG)

   FUNCTION
        This locks several records within a file for exclusive access.
        Timeout is how long to wait in ticks for the records to be available.
        The wait is applied to each attempt to lock each record in the list.
        It is recommended that you always lock a set of records in the same
        order to reduce possibilities of deadlock.

        The array of RecordLock structures is terminated by an entry with
        rec_FH of NULL.

   INPUTS
        record_array - List of records to be locked
        timeout      - Timeout interval.  0 is legal

   RESULT
        success      - Success or failure

   BUGS
        See LockRecord()

   SEE ALSO
        LockRecord(), UnLockRecord(), UnLockRecords()
```

```
dos.library/MakeDosEntry                              dos.library/MakeDosEntry

   NAME
       MakeDosEntry -- Creates a DosList structure (V36)

   SYNOPSIS
       newdlist = MakeDosEntry(name, type)
       D0                      D1    D2

       struct DosList *MakeDosEntry(UBYTE *, LONG)

   FUNCTION
       Create a DosList structure, including allocating a name and correctly
       null-terminating the BSTR.  It also sets the dol_Type field, and sets
       all other fields to 0.  This routine should be eliminated and replaced
       by a value passed to AllocDosObject()!

   INPUTS
       name - name for the device/volume/assign node.
       type - type of node.

   RESULT
       newdlist - The new device entry or NULL.

   SEE ALSO
       AddDosEntry(), RemDosEntry(), FindDosEntry(), LockDosList(),
       NextDosEntry(), FreeDosEntry()
```

```
dos.library/MakeLink                                      dos.library/MakeLink

   NAME
       MakeLink -- Creates a filesystem link (V36)

   SYNOPSIS
       success = MakeLink( name, dest, soft )
       D0                  D1    D2    D3

       BOOL MakeLink( UBYTE *, LONG, LONG )

   FUNCTION
       Create a filesystem link from 'name' to dest.  For "soft-links",
       dest is a pointer to a null-terminated path string.  For "hard-
       links", dest is a lock (BPTR).  'soft' is FALSE for hard-links,
       non-zero otherwise.

       Soft-links are resolved at access time by a combination of the
       filesystem (by returning ERROR_IS_SOFT_LINK to dos), and by
       Dos (using ReadLink() to resolve any links that are hit).

       Hard-links are resolved by the filesystem in question.  A series
       of hard-links to a file are all equivalent to the file itself.
       If one of the links (or the original entry for the file) is
       deleted, the data remains until there are no links left.

   INPUTS
       name - Name of the link to create
       dest - CPTR to path string, or BPTR lock
       soft - FALSE for hard-links, non-zero for soft-links

   RESULT
       Success - boolean

   BUGS
       In V36, soft-links didn't work in the ROM filesystem.  This was
       fixed for V37.

   SEE ALSO
       ReadLink(), Open(), Lock()
```

dos.library Page 87

```
dos.library/MatchEnd                                     dos.library/MatchEnd

   NAME
       MatchEnd -- Free storage allocated for MatchFirst()/MatchNext()  (V36)

   SYNOPSIS
       MatchEnd(AnchorPath)
                   D1

       VOID MatchEnd(struct AnchorPath *)

   FUNCTION
       Return all storage associated with a given search.

   INPUTS
       AnchorPath - Anchor used for MatchFirst()/MatchNext()
                    MUST be longword aligned!

   SEE ALSO
       MatchFirst(), ParsePattern(), Examine(), CurrentDir(), Examine(),
       MatchNext(), ExNext(), <dos/dosasl.h>
```

dos.library Page 88

```
dos.library/MatchFirst                                 dos.library/MatchFirst

   NAME
       MatchFirst -- Finds file that matches pattern (V36)

   SYNOPSIS
       error = MatchFirst(pat, AnchorPath)
       D0                 D1       D2

       BOOL MatchFirst(UBYTE *, struct AnchorPath *)

   FUNCTION
       Locates the first file or directory that matches a given pattern.
       MatchFirst() is passed your pattern (you do not pass it through
       ParsePattern() - MatchFirst() does that for you), and the control
       structure.  MatchFirst() normally initializes your AnchorPath
       structure for you, and returns the first file that matched your
       pattern, or an error.  Note that MatchFirst()/MatchNext() are unusual
       for Dos in that they return 0 for success, or the error code (see
       <dos/dos.h>), instead of the application getting the error code
       from IoErr().

       When looking at the result of MatchFirst()/MatchNext(), the ap_Info
       field of your AnchorPath has the results of an Examine() of the object.
       You normally get the name of the object from fib_FileName, and the
       directory it's in from ap_Current->an_Lock.  To access this object,
       normally you would temporarily CurrentDir() to the lock, do an action
       to the file/dir, and then CurrentDir() back to your original directory.
       This makes certain you affect the right object even when two volumes
       of the same name are in the system.  You can use ap_Buf (with
       ap_Strlen) to get a name to report to the user.

       See ParsePattern() for more information on the patterns.

   INPUTS
       pat        - Pattern to search for
       AnchorPath - Place holder for search.  MUST be longword aligned!

   RESULT
       error - 0 for success or error code.  (Opposite of most Dos calls)

   SEE ALSO
       MatchNext(), ParsePattern(), Examine(), CurrentDir(), Examine(),
       MatchEnd(), ExNext(), <dos/dosasl.h>
```

```
dos.library/MatchNext                                   dos.library/MatchNext

   NAME
        MatchNext - Finds the next file or directory that matches pattern (V36)

   SYNOPSIS
        error = MatchNext(AnchorPath)
        D0                D1

        BOOL MatchNext(struct AnchorPath *)

   FUNCTION
        Locates the next file or directory that matches a given pattern.
        See <dos/dosasl.h> for more information.  Various bits in the flags
        allow the application to control the operation of MatchNext().

   INPUTS
        AnchorPath - Place holder for search.  MUST be longword aligned!

   RESULT
        error - 0 for success or error code.  (Opposite of most Dos calls)

   SEE ALSO
        MatchFirst(), ParsePattern(), Examine(), CurrentDir(), Examine(),
        MatchEnd(), ExNext(), <dos/dosasl.h>
```

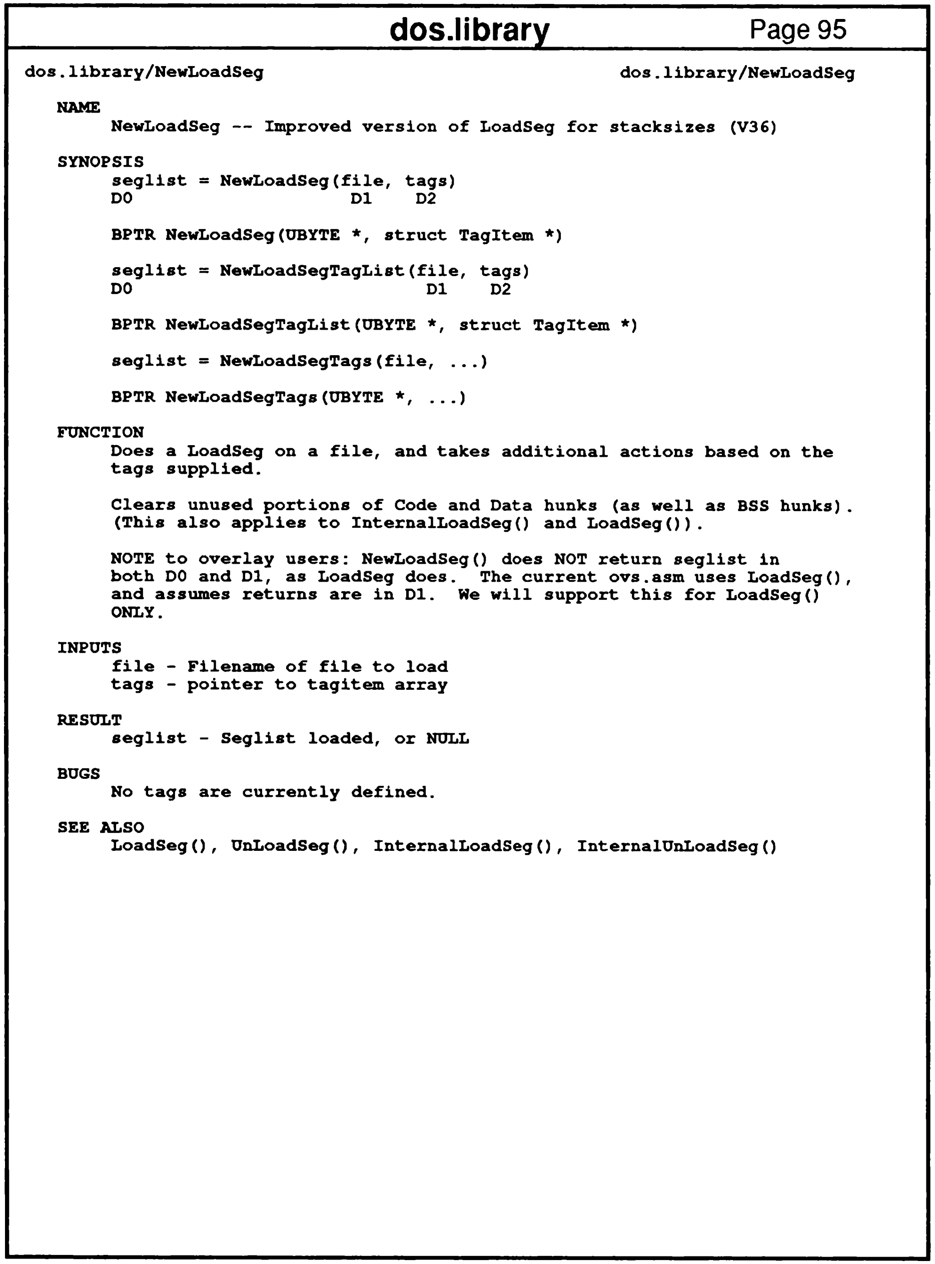
```
dos.library/MatchPattern                                dos.library/MatchPattern

   NAME
        MatchPattern --  Checks for a pattern match with a string (V36)

   SYNOPSIS
        match = MatchPattern(pat, str)
        D0                   D1   D2

        BOOL MatchPattern(UBYTE *, UBYTE *)

   FUNCTION
        Checks for a pattern match with a string.  The pattern must be a
        tokenized string output by ParsePattern().  This routine is
        case-sensitive.

        NOTE: this routine is highly recursive.  You must have at least
        1500 free bytes of stack to call this (it will cut off it's
        recursion before going any deeper than that and return failure).
        That's _currently_ enough for about 100 levels deep of #, (, ~, etc.

   INPUTS
        pat - Special pattern string to match as returned by ParsePattern()
        str - String to match against given pattern

   RESULT
        match - success or failure of pattern match.  On failure,
                IoErr() will return 0 or ERROR_TOO_MANY_LEVELS (starting
                with V37 - before that there was no stack checking).

   SEE ALSO
        ParsePattern(), MatchPatternNoCase(), MatchFirst(), MatchNext()
```

```
dos.library/MatchPatternNoCase                     dos.library/MatchPatternNoCase

   NAME
        MatchPatternNoCase --  Checks for a pattern match with a string (V36)

   SYNOPSIS
        match = MatchPatternNoCase(pat, str)
        D0                         D1   D2

        BOOL MatchPatternNoCase(UBYTE *, UBYTE *)

   FUNCTION
        Checks for a pattern match with a string.  The pattern must be a
        tokenized string output by ParsePatternNoCase().  This routine is
        case-insensitive.

        NOTE: this routine is highly recursive.  You must have at least
        1500 free bytes of stack to call this (it will cut off it's
        recursion before going any deeper than that and return failure).
        That's _currently_ enough for about 100 levels deep of #, (, ~, etc.

   INPUTS
        pat - Special pattern string to match as returned by ParsePatternNoCase()
        str - String to match against given pattern

   RESULT
        match - success or failure of pattern match.  On failure,
                IoErr() will return 0 or ERROR_TOO_MANY_LEVELS (starting
                with V37 - before that there was no stack checking).

   SEE ALSO
        ParsePatternNoCase(), MatchPattern(), MatchFirst(), MatchNext()
```

```
dos.library/MaxCli                                               dos.library/MaxCli

   NAME
        MaxCli -- returns the highest CLI process number possibly in use (V36)

   SYNOPSIS
        number = MaxCli()
        D0

        LONG MaxCli(void)

   FUNCTION
        Returns the highest CLI number that may be in use.  CLI numbers are
        reused, and are usually as small as possible.  To find all CLIs, scan
        using FindCliProc() from 1 to MaxCLI().  The number returned by
        MaxCli() may change as processes are created and destroyed.

   RESULT
        number - The highest CLI number that _may_ be in use.

   SEE ALSO
        FindCliProc(), Cli()
```

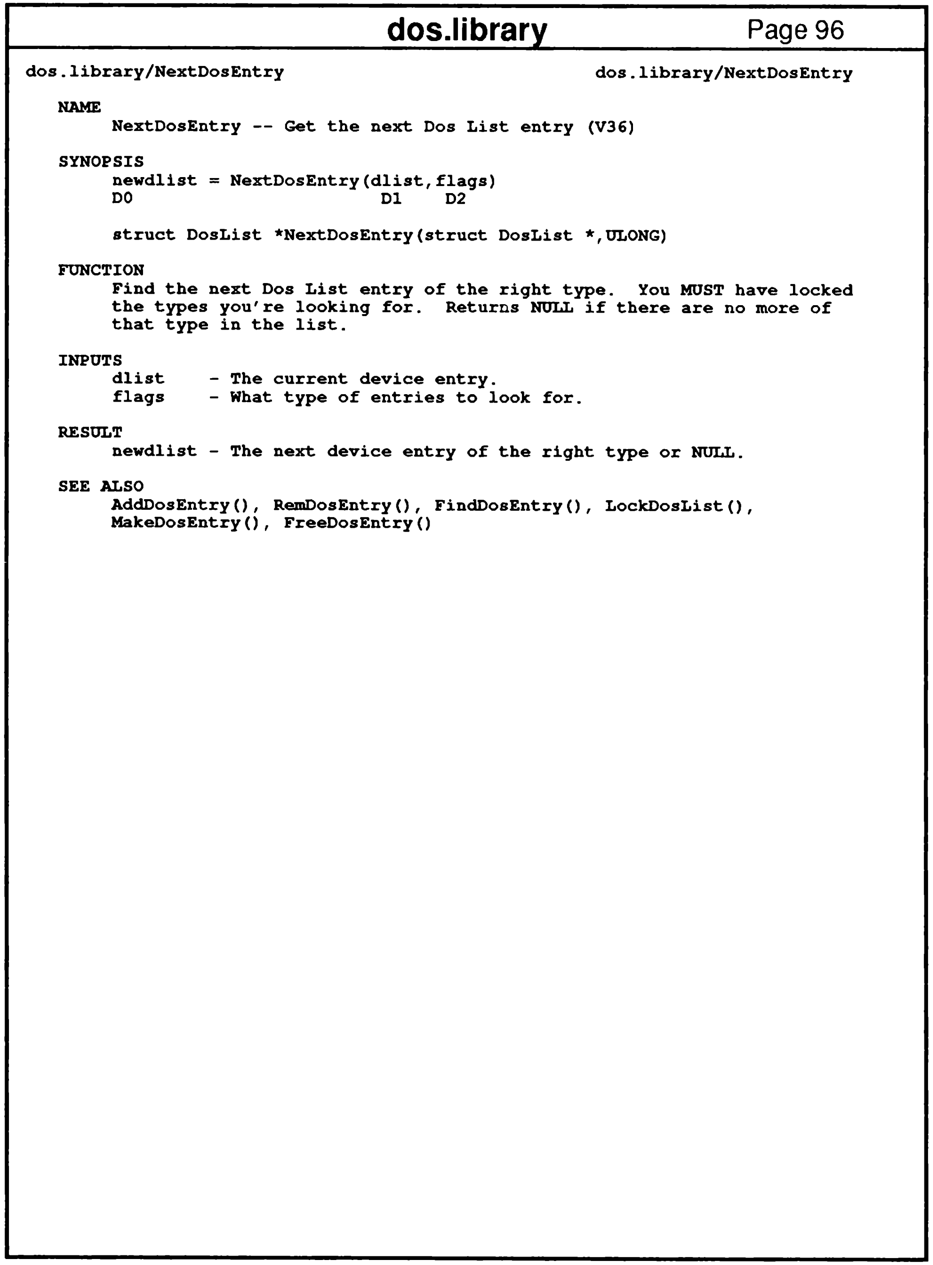
```
dos.library/NameFromFH                                  dos.library/NameFromFH

   NAME
       NameFromFH -- Get the name of an open filehandle (V36)

   SYNOPSIS
       success = NameFromFH(fh, buffer, len)
       D0                   D1    D2     D3

       BOOL NameFromFH(BPTR, char *, LONG)

   FUNCTION
       Returns a fully qualified path for the filehandle.  This routine is
       guaranteed not to write more than len characters into the buffer.  The
       name will be null-terminated.  See NameFromLock() for more information.

   INPUTS
       fh     - Lock of object to be examined.
       buffer - Buffer to store name.
       len    - Length of buffer.

   RESULT
       success - Success/failure indicator.

   SEE ALSO
       NameFromLock()
```

```
dos.library/NameFromLock                              dos.library/NameFromLock

   NAME
       NameFromLock -- Returns the name of a locked object (V36)

   SYNOPSIS
       success = NameFromLock(lock, buffer, len)
       D0                     D1     D2     D3

       BOOL NameFromLock(BPTR, char *, LONG)

   FUNCTION
       Returns a fully qualified path for the lock.  This routine is
       guaranteed not to write more than len characters into the buffer.  The
       name will be null-terminated.  NOTE: if the volume is not mounted,
       the system will request it (unless of course you set pr_WindowPtr to
       -1).  If the volume is not mounted or inserted, it will return an
       error.  If the lock passed in is NULL, "SYS:" will be returned. If
       the buffer is too short, an error will be returned, and IoErr() will
       return ERROR_LINE_TOO_LONG.

   INPUTS
       lock   - Lock of object to be examined.
       buffer - Buffer to store name.
       len    - Length of buffer.

   RESULT
       success - Success/failure indicator.

   BUGS
       Should return the name of the boot volume instead of SYS: for a NULL
       lock.

   SEE ALSO
       NameFromFH(), Lock()
```



dos.library Page 95

```
dos.library/NewLoadSeg                                dos.library/NewLoadSeg

   NAME
       NewLoadSeg -- Improved version of LoadSeg for stacksizes (V36)

   SYNOPSIS
       seglist = NewLoadSeg(file, tags)
       D0                   D1    D2

       BPTR NewLoadSeg(UBYTE *, struct TagItem *)

       seglist = NewLoadSegTagList(file, tags)
       D0                          D1    D2

       BPTR NewLoadSegTagList(UBYTE *, struct TagItem *)

       seglist = NewLoadSegTags(file, ...)

       BPTR NewLoadSegTags(UBYTE *, ...)

   FUNCTION
       Does a LoadSeg on a file, and takes additional actions based on the
       tags supplied.

       Clears unused portions of Code and Data hunks (as well as BSS hunks).
       (This also applies to InternalLoadSeg() and LoadSeg()).

       NOTE to overlay users: NewLoadSeg() does NOT return seglist in
       both D0 and D1, as LoadSeg does.  The current ovs.asm uses LoadSeg(),
       and assumes returns are in D1.  We will support this for LoadSeg()
       ONLY.

   INPUTS
       file - Filename of file to load
       tags - pointer to tagitem array

   RESULT
       seglist - Seglist loaded, or NULL

   BUGS
       No tags are currently defined.

   SEE ALSO
       LoadSeg(), UnLoadSeg(), InternalLoadSeg(), InternalUnLoadSeg()
```



dos.library Page 96

```
dos.library/NextDosEntry                                dos.library/NextDosEntry

   NAME
       NextDosEntry -- Get the next Dos List entry (V36)

   SYNOPSIS
       newdlist = NextDosEntry(dlist,flags)
       D0                      D1    D2

       struct DosList *NextDosEntry(struct DosList *,ULONG)

   FUNCTION
       Find the next Dos List entry of the right type.  You MUST have locked
       the types you're looking for.  Returns NULL if there are no more of
       that type in the list.

   INPUTS
       dlist    - The current device entry.
       flags    - What type of entries to look for.

   RESULT
       newdlist - The next device entry of the right type or NULL.

   SEE ALSO
       AddDosEntry(), RemDosEntry(), FindDosEntry(), LockDosList(),
       MakeDosEntry(), FreeDosEntry()
```

```
dos.library/Open                                              dos.library/Open

    NAME
        Open -- Open a file for input or output

    SYNOPSIS
        file = Open( name, accessMode )
        D0           D1    D2

        BPTR Open(char *, LONG)

    FUNCTION
        The named file is opened and a file handle returned.  If the
        accessMode is MODE_OLDFILE, an existing file is opened for reading
        or writing. If the value is MODE_NEWFILE, a new file is created for
        writing. MODE_READWRITE opens a file with an shared lock, but
        creates it if it didn't exist.  Open types are documented in the
        <dos/dos.h> or <libraries/dos.h> include file.

        The 'name' can be a filename (optionally prefaced by a device
        name), a simple device such as NIL:, a window specification such as
        CON: or RAW: followed by window parameters, or "*", representing the
        current window.  Note that as of V36, "*" is obsolete, and CONSOLE:
        should be used instead.

        If the file cannot be opened for any reason, the value returned
        will be zero, and a secondary error code will be available by
        calling the routine IoErr().

    INPUTS
        name       - pointer to a null-terminated string
        accessMode - integer

    RESULTS
        file - BCPL pointer to a file handle

    SEE ALSO
        Close(), ChangeMode(), NameFromFH(), ParentOfFH(), ExamineFH()
```

```
dos.library/OpenFromLock                              dos.library/OpenFromLock

   NAME
        OpenFromLock -- Opens a file you have a lock on (V36)

   SYNOPSIS
        fh = OpenFromLock(lock)
        D0                D1

        BPTR OpenFromLock(BPTR)

   FUNCTION
        Given a lock, this routine performs an open on that lock.  If the open
        succeeds, the lock is (effectively) relinquished, and should not be
        UnLock()ed or used.  If the open fails, the lock is still usable.
        The lock associated with the file internally is of the same access
        mode as the lock you gave up - shared is similar to MODE_OLDFILE,
        exclusive is similar to MODE_NEWFILE.

   INPUTS
        lock - Lock on object to be opened.

   RESULT
        fh   - Newly opened file handle or NULL for failure

   BUGS
        In the original V36 autodocs, this was shown (incorrectly) as
        taking a Mode parameter as well.  The prototypes and pragmas were
        also wrong.

   SEE ALSO
        Open(), Close(), Lock(), UnLock()
```

```
dos.library/Output                                       dos.library/Output

    NAME
        Output -- Identify the programs' initial output file handle

    SYNOPSIS
        file = Output()
        D0

        BPTR Output(void)

    FUNCTION
        Output() is used to identify the initial output stream allocated
        when the program was initiated.  Never close the filehandle returned
        by Output().

    RESULTS
        file - BCPL pointer to a file handle

    SEE ALSO
        Input()
```

```
dos.library/ParentDir                                 dos.library/ParentDir

    NAME
        ParentDir -- Obtain the parent of a directory or file

    SYNOPSIS
        newlock = ParentDir( lock )
        D0                   D1

        BPTR ParentDir(BPTR)

    FUNCTION
        The argument 'lock' is associated with a given file or directory.
        ParentDir() returns 'newlock' which is associated the parent
        directory of 'lock'.

        Taking the ParentDir() of the root of the current filing system
        returns a NULL (0) lock.  Note this 0 lock represents the root of
        file system that you booted from (which is, in effect, the parent
        of all other file system roots.)

    INPUTS
        lock - BCPL pointer to a lock

    RESULTS
        newlock - BCPL pointer to a lock

    SEE ALSO
        Lock(), DupLock(), UnLock(), ParentOfFH(), DupLockFromFH()
```

```
dos.library/ParentOfFH                                  dos.library/ParentOfFH

   NAME
        ParentOfFH -- returns a lock on the parent directory of a file (V36)

   SYNOPSIS
        lock = ParentOfFH(fh)
        D0                D1

        BPTR ParentOfFH(BPTR)

   FUNCTION
        Returns a shared lock on the parent directory of the filehandle.

   INPUTS
        fh   - Filehandle you want the parent of.

   RESULT
        lock - Lock on parent directory of the filehandle or NULL for failure.

   SEE ALSO
        Parent(), Lock(), UnLock() DupLockFromFH()
```

```
dos.library/ParsePattern                              dos.library/ParsePattern

   NAME
        ParsePattern -- Create a tokenized string for MatchPattern() (V36)

   SYNOPSIS
        IsWild = ParsePattern(Source, Dest, DestLength)
        d0                    D1      D2    D3

        LONG ParsePattern(UBYTE *, UBYTE *, LONG)

   FUNCTION
        Tokenizes a pattern, for use by MatchPattern().  Also indicates if
        there are any wildcards in the pattern (i.e. whether it might match
        more than one item).  Note that Dest must be at least 2 times as
        large as Source plus bytes to be (almost) 100% certain of no
        buffer overflow.  This is because each input character can currently
        expand to 2 tokens (with one exception that can expand to 3, but
        only once per string).  Note: this implementation may change in
        the future, so you should handle error returns in all cases, but
        the size above should still be a reasonable upper bound for a buffer
        allocation.

        The patterns are fairly extensive, and approximate some of the ability
        of Unix/grep "regular expression" patterns.  Here are the available
        tokens:

        ?       Matches a single character.
        #       Matches the following expression 0 or more times.
        (ab|cd) Matches any one of the items seperated by '|'.
        ~       Negates the following expression.  It matches all strings
                that do not match the expression (aka ~(foo) matches all
                strings that are not exactly "foo").
        [abc]   Character class: matches any of the characters in the class.
        [~bc]   Character class: matches any of the characters not in the
                class.
        a-z     Character range (only within character classes).
        %       Matches 0 characters always (useful in "(foo|bar|%)").
        *       Synonym for "#?", not available by default in 2.0.  Available
                as an option that can be turned on.

        "Expression" in the above table means either a single character
        (ex: "#?"), or an alternation (ex: "#(ab|cd|ef)"), or a character
        class (ex: "#[a-zA-Z]").

   INPUTS
        source     - unparsed wildcard string to search for.
       dest       - output string, gets tokenized version of input.
        DestLength - length available in destination (should be at least as
                     twice as large as source + 2 bytes).

   RESULT
        IsWild - 1 means there were wildcards in the pattern,
                 0 means there were no wildcards in the pattern,
                -1 means there was a buffer overflow or other error

   BUGS
        Should set IoErr() to something useful (not currently set) on an
        error.

   SEE ALSO
        ParsePatternNoCase(), MatchPattern(), MatchFirst(), MatchNext()
```

dos.library Page 103

```
dos.library/ParsePatternNoCase                 dos.library/ParsePatternNoCase

   NAME
        ParsePatternNoCase -- Create a tokenized string for
                                        MatchPatternNoCase() (V36)

   SYNOPSIS
        IsWild = ParsePatternNoCase(Source, Dest, DestLength)
        d0                          D1      D2    D3

        LONG ParsePatternNoCase(UBYTE *, UBYTE *, LONG)

   FUNCTION
        Tokenizes a pattern, for use by MatchPatternNoCase().  Also indicates
        if there are any wildcards in the pattern (i.e. whether it might match
        more than one item).  Note that Dest must be at least 2 times as
        large as Source plus 2 bytes.

        For a description of the wildcards, see ParsePattern().

   INPUTS
        source     - unparsed wildcard string to search for.
       dest       - output string, gets tokenized version of input.
        DestLength - length available in destination (should be at least as
                     twice as large as source + 2 bytes).

   RESULT
        IsWild - 1 means there were wildcards in the pattern,
                 0 means there were no wildcards in the pattern,
                -1 means there was a buffer overflow or other error

   BUGS
        Should set IoErr() to something useful (not currently set) on an
        error.

   SEE ALSO
        ParsePattern(), MatchPatternNoCase(), MatchFirst(), MatchNext()
```

dos.library Page 104

```
dos.library/PathPart                                     dos.library/PathPart

   NAME
        PathPart -- Returns a pointer to the end of the next-to-last (V36)
                    component of a path.

   SYNOPSIS
        fileptr = PathPart( path )
        D0                  D1

        UBYTE *PathPart( UBYTE * )

   FUNCTION
        This function returns a pointer to the character after the next-to-last
        component of a path specification, which will normally be the directory
        name.  If there is only one component, it returns a pointer to the
        beginning of the string.  The only real difference between this and
        FilePart() is the handling of '/'.

   INPUTS
        path - pointer to an path string.  May be relative to the current
               directory or the current disk.

   RESULT
        fileptr - pointer to the end of the next-to-last component of the path.

   EXAMPLE
        PathPart("xxx:yyy/zzz/qqq") would return a pointer to the last '/'.
        PathPart("xxx:yyy") would return a pointer to the first 'y').

   SEE ALSO
        FilePart(), AddPart()
```

```
dos.library/PrintFault                                dos.library/PrintFault

   NAME
       PrintFault -- Returns the text associated with a DOS error code (V36)

   SYNOPSIS
       success = PrintFault(code, header)
       D0                   D1     D2

       BOOL PrintFault(LONG, UBYTE *)

   FUNCTION
       This routine obtains the error message text for the given error code.
       This is similar to the Fault() function, except that the output is
       written to the default output channel with buffered output.
       The value returned by IoErr() is set to the code passed in.

   INPUTS
       code   - Error code
       header - header to output before error text

   RESULT
       success - Success/failure code.

   SEE ALSO
       IoErr(), Fault(), SetIoErr(), Output(), FPuts()
```

```
dos.library/PutStr                                         dos.library/PutStr

   NAME
       PutStr -- Writes a string the the default output (buffered) (V36)

   SYNOPSIS
       error = PutStr(str)
       D0             D1

       LONG PutStr(UBYTE *)

   FUNCTION
       This routine writes an unformatted string to the default output.  No
       newline is appended to the string and any error is returned.  This
       routine is buffered.

   INPUTS
       str   - Null-terminated string to be written to default output

   RESULT
       error - 0 for success, -1 for any error.  NOTE: this is opposite
               most Dos function returns!

   SEE ALSO
       FPuts(), FPutC(), FWrite()
```

```
dos.library/Read                                        dos.library/Read

   NAME
       Read -- Read bytes of data from a file

   SYNOPSIS
       actualLength = Read( file, buffer, length )
       D0                   D1    D2      D3

       LONG Read(BPTR, void *, LONG)

   FUNCTION
       Data can be copied using a combination of Read() and Write().
       Read() reads bytes of information from an opened file (represented
       here by the argument 'file') into the buffer given. The argument
       'length' is the length of the buffer given.

       The value returned is the length of the information actually read.
       So, when 'actualLength' is greater than zero, the value of
       'actualLength' is the the number of characters read. Usually Read
       will try to fill up your buffer before returning. A value of zero
       means that end-of-file has been reached. Errors are indicated by a
       value of -1.

       Note: this is an unbuffered routine (the request is passed directly
       to the filesystem.)  Buffered I/O is more efficient for small
       reads and writes; see FGetC().

   INPUTS
       file - BCPL pointer to a file handle
       buffer - pointer to buffer
       length - integer

   RESULTS
       actualLength - integer

   SEE ALSO
       Open(), Close(), Write(), Seek(), FGetC()
```

```
dos.library/ReadArgs                                    dos.library/ReadArgs

   NAME
       ReadArgs - Parse the command line input (V36)

   SYNOPSIS
       result = ReadArgs(template, array, rdargs)
       D0                D1        D2     D3

       struct RDArgs * ReadArgs(UBYTE *, LONG *, struct RDArgs *)

   FUNCTION
       Parses and argument string according to a template.  Normally gets
       the arguments by reading buffered IO from Input(), but also can be
       made to parse a string.  MUST be matched by a call to FreeArgs().

       ReadArgs() parses the commandline according to a template that is
       passed to it.  This specifies the different command-line options and
       their types.  A template consists of a list of options.  Options are
       named in "full" names where possible (for example, "Quick" instead of
       "Q").  Abbreviations can also be specified by using "abbrev=option"
       (for example, "Q=Quick").

       Options in the template are separated by commas.  To get the results
       of ReadArgs(), you examine the array of longwords you passed to it
       (one entry per option in the template).  This array should be cleared
       (or initialized to your default values) before passing to ReadArgs().
       Exactly what is put in a given entry by ReadArgs() depends on the type
       of option.  The default is a string (a sequence of non-whitespace
       characters, or delimited by quotes, which will be stripped by
       ReadArgs()), in which case the entry will be a pointer.

       Options can be followed by modifiers, which specify things such as
       the type of the option.  Modifiers are specified by following the
       option with a '/' and a single character modifier.  Multiple modifiers
       can be specified by using multiple '/'s.  Valid modifiers are:

       /S - Switch.  This is considered a boolean variable, and will be
            set if the option name appears in the command-line.  The entry
            is the boolean (0 for not set, non-zero for set).

       /K - Keyword.  This means that the option will not be filled unless
            the keyword appears.  For example if the template is "Name/K",
            then unless "Name=<string>" or "Name <string>" appears in the
            command line, Name will not be filled.

       /N - Number.  This parameter is considered a decimal number, and will
            be converted by ReadArgs.  If an invalid number is specified,
            an error will be returned.  The entry will be a pointer to the
            longword number (this is how you know if a number was specified).

       /T - Toggle.  This is similar to a switch, but when specified causes
            the boolean value to "toggle".  Similar to /S.

       /A - Required.  This keyword must be given a value during command-line
            processing, or an error is returned.

       /F - Rest of line.  If this is specified, the entire rest of the line
            is taken as the parameter for the option, even if other option
            keywords appear in it.

       /M - Multiple strings.  This means the argument will take any number
            of strings, returning them as an array of strings.  Any arguments
            not considered to be part of another option will be added to this
            option.  Only one /M should be specified in a template.  Example:
            for a template "Dir/M,All/S" the command-line "foo bar all qwe"
```

```
        will set the boolean "all", and return an array consisting of
        "foo", "bar", and "qwe".  The entry in the array will be a pointer
        to an array of string pointers, the last of which will be NULL.

        There is an interaction between /M parameters and /A parameters.
        If there are unfilled /A parameters after parsing, it will grab
        strings from the end of a previous /M parameter list to fill the
        /A's.  This is used for things like Copy ("From/A/M,To/A").

    ReadArgs() returns a struct RDArgs if it succeeds.  This serves as an
    "anchor" to allow FreeArgs() to free the associated memory.  You can
    also pass in a struct RDArgs to control the operation of ReadArgs()
    (normally you pass NULL for the parameter, and ReadArgs() allocates
    one for you).  This allows providing different sources for the
    arguments, providing your own string buffer space for temporary
    storage, and extended help text.  See <dos/rdargs.h> for more
    information on this.  Note: if you pass in a struct RDArgs, you must
    still call FreeArgs() to release storage that gets attached to it,
    but you are responsible for freeing the RDArgs yourself.

INPUTS
    template - formatting string
    array    - array of longwords for results, 1 per template entry
    rdargs   - optional rdargs structure for options.  AllocDosObject
               should be used for allocating them if you pass one in.

RESULT
    result   - a struct RDArgs or NULL for failure.

BUGS
    In V36, there were a couple of minor bugs with certain argument
    combinations (/M/N returned strings, /T didn't work, and /K and
    /F interacted).  These should be fixed for V37.

SEE ALSO
    FindArg(), ReadItem(), FreeArgs(), AllocDosObject()
```

```
dos.library/ReadItem                                   dos.library/ReadItem

  NAME
    ReadItem - reads a single argument/name from command line (V36)

  SYNOPSIS
    value = ReadItem(buffer, maxchars, input)
    D0               D1      D2        D3

    LONG ReadItem(UBYTE *, LONG, struct CHSource *)

  FUNCTION
    Reads a "word" from either Input() (buffered), or via CHSource, if it
    is non-NULL (see <dos/rdargs.h> for more information).  Handles
    quoting and some '*' substitutions (*e and *n) inside quotes (only).
    See dos/dos.h for a listing of values returned by ReadItem()
    (ITEM_XXXX).  A "word" is delimited by whitespace, quotes, or an EOF.

    ReadItem always unreads the last thing read (UnGetC(fh,-1)) so the
    caller can find out what the terminator was.

  INPUTS
    buffer   - buffer to store word in.
    maxchars - size of the buffer
    input    - CHSource input or NULL (uses FGetC(Input()))

  RESULT
    value - See <dos/dos.h> for return values.

  SEE ALSO
    ReadArgs(), FindArg(), UnGetC(), FGetC(), Input(), <dos/dos.h>,
    <dos/rdargs.h>, FreeArgs()
```

dos.library Page 111

```
dos.library/ReadLink                                     dos.library/ReadLink

   NAME
        ReadLink -- Reads the path for a soft filesystem link (V36)

   SYNOPSIS
        success = ReadLink( port, lock, path, buffer, size)
        D0                  D1    D2    D3    D4      D5

        BOOL ReadLink( struct MsgPort *, BPTR, UBYTE *, UBYTE *, ULONG)

   FUNCTION
        ReadLink() takes a lock/name pair (usually from a failed attempt
        to use them to access an object with packets), and asks the
        filesystem to find the softlink and fill buffer with the modified
        path string.  You then start the resolution process again by
        calling GetDeviceProc() with the new string from ReadLink().

        Soft-links are resolved at access time by a combination of the
        filesystem (by returning ERROR_IS_SOFT_LINK to dos), and by
        Dos (using ReadLink() to resolve any links that are hit).

   INPUTS
        port - msgport of the filesystem
        lock - lock this path is relative to on the filesystem
        path - path that caused the ERROR_IS_SOFT_LINK
        buffer - pointer to buffer for new path from handler.
        size - size of buffer.

   RESULT
        Success - boolean

   BUGS
        In V36, soft-links didn't work in the ROM filesystem.  This was
        fixed for V37.

   SEE ALSO
        MakeLink(), Open(), Lock(), GetDeviceProc()
```

dos.library Page 112

```
dos.library/Relabel                                         dos.library/Relabel

   NAME
        Relabel -- Change the volume name of a volume (V36)

   SYNOPSIS
        success = Relabel(volumename,name)
        D0                D1         D2

        BOOL Relabel(char *,char *)

   FUNCTION
        Changes the volumename of a volume, if supported by the filesystem.

   INPUTS
        volumename - Full name of device to rename (with ':')
        newname    - New name to apply to device (without ':')

   RESULT
        success    - Success/failure indicator

   SEE ALSO
```

```
dos.library/RemAssignList                          dos.library/RemAssignList

   NAME
       RemAssignList -- Remove an entry from a multi-dir assign (V36)

   SYNOPSIS
       success = RemAssignList(name,lock)
       D0                      D1   D2

       BOOL RemAssignList(char *,BPTR)

   FUNCTION
       Removes an entry from a multi-directory assign.  The entry removed is
       the first one for which SameLock with 'lock' returns that they are on
       the same object.  The lock for the entry in the list is unlocked (not
       the entry passed in).

   INPUTS
       name - Name of device to remove lock from (without trailing ':')
       lock - Lock associated with the object to remove from the list

   RESULT
       success - Success/failure indicator.

   SEE ALSO
       Lock(), AssignLock(), AssignPath(), AssignLate(), DupLock(),
       AssignAdd(), UnLock()
```

```
dos.library/RemDosEntry                              dos.library/RemDosEntry

   NAME
       RemDosEntry -- Removes a Dos List entry from it's list (V36)

   SYNOPSIS
       success = RemDosEntry(dlist)
       D0                    D1

       BOOL RemDosEntry(struct DosList *)

   FUNCTION
       This removes an entry from the Dos Device list.  The memory associated
       with the entry is NOT freed.  NOTE: you must have locked the Dos List
       with the appropriate flags before calling this routine.

   INPUTS
       dlist   - Device list entry to be removed.

   RESULT
       success - Success/failure indicator

   SEE ALSO
       AddDosEntry(), FindDosEntry(), NextDosEntry(), LockDosList(),
       MakeDosEntry(), FreeDosEntry()
```

dos.library Page 115

```
dos.library/RemSegment                                 dos.library/RemSegment

   NAME
        RemSegment - Removes a resident segment from the resident list (V36)

   SYNOPSIS
        success = RemSegment(segment)
        D0                   D1

        BOOL RemSegment(struct Segment *)

   FUNCTION
        Removes a resident segment from the Dos resident segment list,
        unloads it, and does any other cleanup required.  Will only succeed
        if the seg_UC (usecount) is 0.

   INPUTS
        segment - the segment to be removed

   RESULT
        success - success or failure.

   SEE ALSO
        FindSegment(), AddSegment()
```

dos.library Page 116

```
dos.library/Rename                                         dos.library/Rename

    NAME
        Rename -- Rename a directory or file

    SYNOPSIS
        success = Rename( oldName, newName )
        D0                D1       D2

        BOOL Rename(char *, char *)

    FUNCTION
        Rename() attempts to rename the file or directory specified as
        'oldName' with the name 'newName'. If the file or directory
        'newName' exists, Rename() fails and returns an error. Both
        'oldName' and the 'newName' can contain a directory specification.
        In this case, the file will be moved from one directory to another.

        Note: it is impossible to Rename() a file from one volume to
        another.

    INPUTS
        oldName - pointer to a null-terminated string
        newName - pointer to a null-terminated string

    RESULTS
        success - boolean

    SEE ALSO
        Relabel()
```

```
dos.library/ReplyPkt                                         dos.library/ReplyPkt

   NAME
        ReplyPkt -- replies a packet to the person who sent it to you (V36)

   SYNOPSIS
        ReplyPkt(packet, result1, result2)
                   D1      D2       D3

        void ReplyPkt(struct DosPacket *, LONG, LONG)

   FUNCTION
        This returns a packet to the process which sent it to you.  In
        addition, puts your pr_MsgPort address in dp_Port, so using ReplyPkt()
        again will send the message to you.  (This is used in "ping-ponging"
        packets between two processes).  It uses result 1 & 2 to set the
        dp_Res1 and dp_Res2 fields of the packet.

   INPUTS
        packet  - packet to reply, assumed to set up correctly.
        result1 - first result
        result2 - secondary result

   SEE ALSO
        DoPkt(), SendPkt(), WaitPkt(), IoErr()
```

```
dos.library/RunCommand                                     dos.library/RunCommand

   NAME
        RunCommand -- Runs a program using the current process (V36)

   SYNOPSIS
        rc = RunCommand(seglist, stacksize, argptr, argsize)
        D0                D1         D2        D3       D4

        LONG RunCommand(BPTR, ULONG, char *, ULONG)

   FUNCTION
        Runs a command on your process/cli.  Seglist may be any language,
        including BCPL programs.  Stacksize is in bytes.  argptr is a null-
        terminated string, argsize is its length.  Returns the returncode the
        program exited with in d0. Returns -1 if the stack couldn't be
        allocated.

        RunCommand also takes care of setting up the current input filehandle
        in such a way that ReadArgs() can be used in the program, and restores
        the state of the buffering before returning.  It also sets the value
        returned by GetArgStr(), and restores it before returning.  NOTE:
        the setting of the filehandle was added in V37.

   INPUTS
        seglist   - Seglist of command to run.
        stacksize - Number of bytes to allocate for stack space
        argptr    - Pointer to argument command string.
        argsize   - Number of bytes in argument command.

   RESULT
        rc        - Return code from executed command. -1 indicates failure

   SEE ALSO
        CreateNewProc(), SystemTagList(), Execute(), GetArgStr()
```

```
dos.library/SameDevice                                 dos.library/SameDevice

   NAME
        SameDevice -- Are two locks are on partitions of the device? (V37)

   SYNOPSIS
        same = SameDevice(lock1, lock2)
        D0                 D1     D2

        BOOL SameDevice( BPTR, BPTR )

   FUNCTION
        SameDevice() returns whether two locks refer to partitions that
        are on the same physical device (if it can figure it out).  This
        may be useful in writing copy routines to take advantage of
        asynchronous multi-device copies.

        Entry existed in V36 and always returned 0.

   INPUTS
        lock1,lock2 - locks

   RESULT
        same - whether they're on the same device as far as Dos can determine.
```

```
dos.library/SameLock                                       dos.library/SameLock

   NAME
        SameLock -- returns whether two locks are on the same object (V36)

   SYNOPSIS
        value = SameLock(lock1, lock2)
        D0                D1     D2

        LONG SameLock(BPTR, BPTR)

   FUNCTION
        Compares two locks.  Returns LOCK_SAME if they are on the same object,
        LOCK_SAME_HANDLER if on different objects on the same handler, and
        LOCK_DIFFERENT if they are on different handlers.

   INPUTS
        lock1 - 1st lock for comparison
        lock2 - 2nd lock for comparison

   RESULT
        value - LOCK_SAME, LOCK_SAME_HANDLER, or LOCK_DIFFERENT

   BUGS
        Should do more extensive checks for NULL against a real lock, checking
        to see if the real lock is a lock on the root of the boot volume.

   SEE ALSO
```

dos.library Page 121

```
dos.library/Seek                                              dos.library/Seek

    NAME
        Seek -- Set the current position for reading and writing

    SYNOPSIS
        oldPosition = Seek( file, position, mode )
        D0                  D1    D2        D3

        LONG Seek(BPTR, LONG, LONG)

    FUNCTION
        Seek() sets the read/write cursor for the file 'file' to the
        position 'position'. This position is used by both Read() and
        Write() as a place to start reading or writing. The result is the
        current absolute position in the file, or -1 if an error occurs, in
        which case IoErr() can be used to find more information. 'mode' can
        be OFFSET_BEGINNING, OFFSET_CURRENT or OFFSET_END. It is used to
        specify the relative start position. For example, 20 from current
        is a position 20 bytes forward from current, -20 is 20 bytes back
        from current.

        So that to find out where you are, seek zero from current. The end
        of the file is a Seek() positioned by zero from end. You cannot
        Seek() beyond the end of a file.

    INPUTS
        file - BCPL pointer to a file handle
        position - integer
        mode - integer

    RESULTS
        oldPosition - integer

    SEE ALSO
        Read(), Write(), SetFileSize()
```

dos.library Page 122

```
dos.library/SelectInput                                dos.library/SelectInput

    NAME
        SelectInput -- Select a filehandle as the default input channel (V36)

    SYNOPSIS
        old_fh = SelectInput(fh)
        D0                   D1

        BPTR SelectInput(BPTR)

    FUNCTION
        Set the current input as the default input for the process.
        This changes the value returned by Input().  old_fh should
        be closed or saved as needed.

    INPUTS
        fh     - Newly default input handle

    RESULT
        old_fh - Previous default input filehandle

    SEE ALSO
        Input(), SelectOutput(), Output()
```

```
dos.library/SelectOutput                                 dos.library/SelectOutput

   NAME
        SelectOutput -- Select a filehandle as the default input channel (V36)

   SYNOPSIS
        old_fh = SelectOutput(fh)
        D0                    D1

        BPTR SelectOutput(BPTR)

   FUNCTION
        Set the current output as the default output for the process.
        This changes the value returned by Output().  old_fh should
        be closed or saved as needed.

   INPUTS
        fh     - Newly desired output handle

   RESULT
        old_fh - Previous current output

   SEE ALSO
        Output(), SelectInput(), Input()
```

```
dos.library/SendPkt                                           dos.library/SendPkt

   NAME
        SendPkt -- Sends a packet to a handler (V36)

   SYNOPSIS
        SendPkt(packet, port, replyport)
                D1      D2    D3

        void SendPkt(struct DosPacket *,struct MsgPort *,struct MsgPort *)

   FUNCTION
        Sends a packet to a handler and does not wait.  All fields in the
        packet must be initialized before calling this routine.  The packet
        will be returned to replyport.  If you wish to use this with
        WaitPkt(), use the address or your pr_MsgPort for replyport.

   INPUTS
        packet - packet to send, must be initialized and have a message.
        port   - pr_MsgPort of handler process to send to.
        replyport - MsgPort for the packet to come back to.

   SEE ALSO
        DoPkt(), WaitPkt(), AllocDosObject(), FreeDosObject(), AbortPkt()
```

```
dos.library/SetArgStr                                   dos.library/SetArgStr

   NAME
       SetArgStr -- Sets the arguments for the current process (V36)

   SYNOPSIS
       oldptr = SetArgStr(ptr)
       D0                 D1

       UBYTE *SetArgStr(UBYTE *)

   FUNCTION
       Sets the arguments for the current program.  The ptr MUST be reset
       to it's original value before process exit.

   INPUTS
       ptr - pointer to new argument string.

   RESULT
       oldptr - the previous argument string

   SEE ALSO
       GetArgStr(), RunCommand()
```

```
dos.library/SetComment                                  dos.library/SetComment

   NAME
       SetComment -- Change a files' comment string

   SYNOPSIS
       success = SetComment( name, comment )
       D0                    D1    D2

       BOOL SetComment(char *, char *)

   FUNCTION
       SetComment() sets a comment on a file or directory. The comment is
       a pointer to a null-terminated string of up to 80 characters.

   INPUTS
       name    - pointer to a null-terminated string
       comment - pointer to a null-terminated string

   RESULTS
       success - boolean

   SEE ALSO
       Examine(), ExNext(), SetProtection()
```

dos.library

Page 127

```
dos.library/SetConsoleTask                         dos.library/SetConsoleTask

   NAME
       SetConsoleTask -- Sets the default console for the process (V36)

   SYNOPSIS
       oldport = SetConsoleTask(port)
       D0                       D1

       struct MsgPort *SetConsoleTask(struct MsgPort *)

   FUNCTION
       Sets the default console task's port (pr_ConsoleTask) for the
       current process.

   INPUTS
       port - The pr_MsgPort of the default console handler for the process

   RESULT
       oldport - The previous ConsoleTask value.

   SEE ALSO
       GetConsoleTask(), Open()
```

dos.library

Page 128

```
dos.library/SetCurrentDirName                   dos.library/SetCurrentDirName

   NAME
       SetCurrentDirName -- Sets the directory name for the process (V36)

   SYNOPSIS
       success = SetCurrentDirName(name)
       D0                          D1

       BOOL SetCurrentDirName(char *)

   FUNCTION
       Sets the name for the current dir in the cli structure.  If the name
       is too long to fit, a failure is returned, and the old value is left
       intact.  It is advised that you inform the user of this condition.
       This routine is safe to call even if there is no CLI structure.

   INPUTS
       name    - Name of directory to be set.

   RESULT
       success - Success/failure indicator

   BUGS
       This clips to a fixed (1.3 compatible) size.

   SEE ALSO
       GetCurrentDirName()
```

```
dos.library/SetFileDate                               dos.library/SetFileDate

   NAME
        SetFileDate -- Sets the modification date for a file or dir (V36)

   SYNOPSIS
        success = SetFileDate(name, date)
        D0                    D1    D2

        BOOL SetFileDate(char *, struct DateStamp *)

   FUNCTION
        Sets the file date for a file or directory.  Note that for the Old
        File System and the Fast File System, the date of the root directory
        cannot be set.  Other filesystems may not support setting the date
        for all files/directories.

   INPUTS
        name - Name of object
        date - New modification date

   RESULT
        success - Success/failure indication

   SEE ALSO
        DateStamp(), Examine(), ExNext(), ExAll()
```

```
dos.library/SetFileSize                               dos.library/SetFileSize

   NAME
        SetFileSize -- Sets the size of a file (V36)

   SYNOPSIS
        newsize = SetFileSize(fh, offset, mode)
        D0                    D1  D2      D3

        LONG SetFileSize(BPTR, LONG, LONG)

   FUNCTION
        Changes the file size, truncating or extending as needed.  Not all
        handlers may support this; be careful and check the return code.  If
        the file is extended, no values should be assumed for the new bytes.
        If the new position would be before the filehandle's current position
        in the file, the filehandle will end with a position at the
        end-of-file.  If there are other filehandles open onto the file, the
        new size will not leave any filehandle pointing past the end-of-file.
        You can check for this by looking at the new size.

        Do NOT count on any specific values to be in the extended area.

   INPUTS
        fh     - File to be truncated/extended.
        offset - Offset from position determined by mode.
        mode   - One of OFFSET_BEGINNING, OFFSET_CURRENT, or OFFSET_END.

   RESULT
        newsize - position of new end-of-file or -1 for error.

   SEE ALSO
        Seek()
```

```
dos.library/SetFileSysTask                          dos.library/SetFileSysTask

   NAME
       SetFileSysTask -- Sets the default filesystem for the process (V36)

   SYNOPSIS
       oldport = SetFileSysTask(port)
       D0                       D1

       struct MsgPort *SetFileSysTask(struct MsgPort *)

   FUNCTION
       Sets the default filesystem task's port (pr_FileSystemTask) for the
       current process.

   INPUTS
       port - The pr_MsgPort of the default filesystem for the process

   RESULT
       oldport - The previous FileSysTask value

   SEE ALSO
       GetFileSysTask(), Open()
```

```
dos.library/SetIoErr                                      dos.library/SetIoErr

   NAME
       SetIoErr -- Sets the value returned by IoErr() (V36)

   SYNOPSIS
       oldcode = SetIoErr(code)
       D0                 D1

       LONG SetIoErr(LONG);

   FUNCTION
       This routine sets up the secondary result (pr_Result2) return code
       (returned by the IoErr() function).

   INPUTS
       code - Code to be returned by a call to IoErr.

   RESULT
       oldcode - The previous error code.

   SEE ALSO
       IoErr(), Fault(), PrintFault()
```

dos.library

Page 133

```
dos.library/SetMode                                      dos.library/SetMode

   NAME
        SetMode - Set the current behavior of a handler (V36)

   SYNOPSIS
        success = SetMode(fh, mode)
        D0                D1  D2

        BOOL SetMode(BPTR, LONG)

   FUNCTION
        SetMode() sends an ACTION_SCREEN_MODE packet to the handler in
        question, normally for changing a CON: handler to raw mode or
        vice-versa.  For CON:, use 1 to go to RAW: mode, 0 for CON: mode.

   INPUTS
        fh   - filehandle
        mode - The new mode you want

   RESULT
        success - Boolean

   SEE ALSO
```

---

dos.library

Page 134

```
dos.library/SetProgramDir                          dos.library/SetProgramDir

   NAME
        SetProgramDir -- Sets the directory returned by GetProgramDir (V36)

   SYNOPSIS
        oldlock = SetProgramDir(lock)
        D0                      D1

        BPTR SetProgramDir(BPTR)

   FUNCTION
        Sets a shared lock on the directory the program was loaded from.
        This can be used for a program to find data files, etc, that are
        stored with the program, or to find the program file itself.  NULL
        is a valid input.  This can be accessed via GetProgramDir() or
        by using paths relative to PROGDIR:.

   INPUTS
        lock - A lock on the directory the current program was loaded from

   RESULT
        oldlock - The previous ProgramDir.

   SEE ALSO
        GetProgramDir(), Open()
```

dos.library Page 135

```
dos.library/SetProgramName                            dos.library/SetProgramName

   NAME
       SetProgramName -- Sets the name of the program being run (V36)

   SYNOPSIS
       success = SetProgramName(name)
       D0                       D1

       BOOL SetProgramName(char *)

   FUNCTION
       Sets the name for the program in the cli structure.  If the name is
       too long to fit, a failure is returned, and the old value is left
       intact.  It is advised that you inform the user if possible of this
       condition, and/or set the program name to an empty string.
       This routine is safe to call even if there is no CLI structure.

   INPUTS
       name    - Name of program to use.

   RESULT
       success - Success/failure indicator.

   BUGS
       This clips to a fixed (1.3 compatible) size.

   SEE ALSO
       GetProgramName()
```

dos.library Page 136

```
dos.library/SetPrompt                                      dos.library/SetPrompt

   NAME
       SetPrompt -- Sets the CLI/shell prompt for the current process (V36)

   SYNOPSIS
       success = SetPrompt(name)
       D0                  D1

       BOOL SetPrompt(char *)

   FUNCTION
       Sets the text for the prompt in the cli structure.  If the prompt is
       too long to fit, a failure is returned, and the old value is left
       intact.  It is advised that you inform the user of this condition.
       This routine is safe to call even if there is no CLI structure.

   INPUTS
       name    - Name of prompt to be set.

   RESULT
       success - Success/failure indicator.

   BUGS
       This clips to a fixed (1.3 compatible) size.

   SEE ALSO
       GetPrompt()
```

```
dos.library/SetProtection                         dos.library/SetProtection

   NAME
       SetProtection -- Set protection for a file or directory

   SYNOPSIS
       success = SetProtection( name, mask )
       D0                       D1    D2:4

       BOOL SetProtection (char *, LONG)

   FUNCTION
       SetProtection() sets the protection attributes on a file or
       directory. The lower bits of the mask are as follows:

       bit 4: 1 = file has not changed         0 = file has been changed
       bit 3: 1 = reads not allowed,           0 = reads allowed.
       bit 2: 1 = writes not allowed,          0 = writes allowed.
       bit 1: 1 = execution not allowed,       0 = execution allowed.
       bit 0: 1 = deletion not allowed,        0 = deletion allowed.

       Before V36, the Old File System didn't respect the Read and Write
       bits.  In V36 or later and in the FFS, the Read and Write
       bits were respected.

       The archive bit is cleared by the file system whenever the file
       is changed.  Backup utilities will generally set the bit after
       backing up each file.

       The Fast Filing System looks at the read and write bits, and
       the V36 Shell looks at the execute bit, and will refuse to start
       a file as a binary executable if it is set.

       Other bits may will be defined in the <dos/dos.h> include
       files.  Rather than referring to bits by number you should use the
       definitions in <dos/dos.h>.

   INPUTS
       name - pointer to a null-terminated string
       mask - the protection mask required

   RESULTS
       success - boolean

   SEE ALSO
       SetComment(), Examine(), ExNext(), <dos/dos.h>
```

```
dos.library/SetVar                                       dos.library/SetVar

   NAME
       SetVar -- Sets a local or environment variable (V36)

   SYNOPSIS
       success = SetVar( name, buffer, size, flags )
       D0                D1    D2      D3    D4

       BOOL SetVar(UBYTE *, UBYTE *, LONG, ULONG )

   FUNCTION
       Sets a local or environment variable.  It is advised to only use
       ASCII strings inside variables, but not required.

   INPUTS
       name   - pointer to an variable name.  Note variable names follow
                filesystem syntax and semantics.
       buffer - a user allocated area which contains a string that is the
                value to be associated with this variable.
       size   - length of the buffer region in bytes.  -1 means buffer
                contains a null-terminated string.
       flags  - combination of type of var to set (low 8 bits), and
                flags to control the behavior of this routine.  Currently
                defined flags include:

               GVF_LOCAL_ONLY - set a local (to your process) variable.
               GVF_GLOBAL_ONLY - set a global environment variable.

               The default is to set a local environment variable.

   RESULT
       success - If non-zero, the variable was sucessfully set, FALSE
                 indicates failure.

   BUGS
       LV_VAR is the only type that can be global

   SEE ALSO
       GetVar(), DeleteVar(), FindVar(), <dos/var.h>
```

dos.library Page 139

```
dos.library/SetVBuf                                       dos.library/SetVBuf

   NAME
       SetVBuf -- set buffering modes and size (V36)

   SYNOPSIS
       error = SetVBuf(fh, buff, type, size)
       D0              D1   D2    D3    D4

       LONG SetVBuf(BPTR, UBYTE *, LONG, LONG)

   FUNCTION
       Changes the buffering modes and buffer size for a filehandle.
       With buff == NULL, the current buffer will be deallocated and a
       new one of (approximately) size will be allocated.  If buffer is
       non-NULL, it will be used for buffering and must be at least
       max(size,208) bytes long.  If buff is NULL and size is -1,
       then only the buffering mode will be changed.

   INPUTS
       fh   - Filehandle
       buff - buffer pointer for buffered I/O
       type - buffering mode (see <dos/stdio.h>)
       size - size of buffer for buffered I/O (sizes less than 208 bytes
              will be ignored).

   RESULT
       error - 0 if successful.  NOTE: opposite of most dos functions!

   BUGS
       Not implemented yet, always returns 0.

   SEE ALSO
       FputC(), FGetC(), UnGetC(), Flush(), FRead(), FWrite(), FGets(),
       FPuts().
```

dos.library Page 140

```
dos.library/SplitName                                   dos.library/SplitName

   NAME
       SplitName -- splits out a component of a pathname into a buffer (V36)

   SYNOPSIS
       newpos = SplitName(name, separator, buf, oldpos, size)
       D0                 D1    D2         D3   D4      D5

       WORD SplitName(UBYTE *, UBYTE, UBYTE *, WORD, LONG)

   FUNCTION
       This routine splits out the next piece of a name from a given file
       name.  Each piece is copied into the buffer, truncating at size-1
       characters.  The new position is then returned so that it may be
       passed in to the next call to splitname.  If the separator is not
       found within 'size' characters, then size-1 characters plus a null will
       be put into the buffer, and the position of the next separator will
       be returned.

       If a a separator cannot be found, -1 is returned (but the characters
       from the old position to the end of the string are copied into the
       buffer, up to a maximum of size-1 characters).  Both strings are
       null-terminated.

       This function is mainly intended to support handlers.

   INPUTS
       name      - Filename being parsed.
       separator - Separator charactor to split by.
       buf       - Buffer to hold separated name.
       oldpos    - Current position in the file.
       size      - Size of buf in bytes (including null termination);

   RESULT
       newpos    - New position for next call to splitname.  -1 for last one.

   SEE ALSO
       FilePart(), PathPart(), AddPart()
```

```
dos.library/StartNotify                              dos.library/StartNotify

   NAME
       StartNotify -- Starts notification on a file or directory (V36)

   SYNOPSIS
       success = StartNotify(notifystructure)
       D0                       D1

       BOOL StartNotify(struct NotifyRequest *)

   FUNCTION
       Posts a notification request.  Do not modify the notify structure while
       it is active.  You will be notified when the file or directory changes.
       For files, you will be notified after the file is closed.  Not all
       filesystems will support this: applications should NOT require it.  In
       particular, most network filesystems won't support it.

   INPUTS
       notifystructure - A filled-in NotifyRequest structure

   RESULT
       success - Success/failure of request

   BUGS
       The V36 floppy/HD filesystem doesn't actually send notifications.  The
       V36 ram handler (ram:) does.  This has been fixed for V37.

   SEE ALSO
       EndNotify(), <dos/notify.h>
```

```
dos.library/StrToDate                                  dos.library/StrToDate

   NAME
       StrToDate -- Converts a string to a DateStamp (V36)

   SYNOPSIS
       success = StrToDate( datetime )
       D0                   D1

       BOOL StrToDate( struct DateTime * )

   FUNCTION
       Converts a human readable ASCII string into an AmigaDOS
       DateStamp.

   INPUTS
       DateTime - a pointer to an initialized DateTime structure.

       The DateTime structure should   be initialized as follows:

       dat_Stamp  - ignored on input.

       dat_Format - a format   byte which specifies the format of the
               dat_StrDat.  This can   be any of the following (note:
               If value used   is something other than those below,
               the default of FORMAT_DOS is used):

               FORMAT_DOS:       AmigaDOS format (dd-mmm-yy).

               FORMAT_INT:       International format (yy-mmm-dd).

               FORMAT_USA:       American format (mm-dd-yy).

               FORMAT_CDN:       Canadian format (dd-mm-yy).

               FORMAT_DEF:       default format for locale.

       dat_Flags - a flags byte.  The only flag which affects this
                 function is:

               DTF_SUBST:      ignored by this function
               DTF_FUTURE:       If set, indicates that strings such
                                 as (stored in dat_StrDate) "Monday"
                                 refer to "next" monday. Otherwise,
                                 if clear, strings like "Monday"
                                 refer to "last" monday.

       dat_StrDay - ignored bythis function.

       dat_StrDate -   pointer to valid string representing the date.
                 This can be a "DTF_SUBST" style string such as
                 "Today" "Tomorrow" "Monday", or it may be a string
                 as specified by the dat_Format byte.  This will be
                 converted to the ds_Days portion of the DateStamp.
                 If this pointer is NULL, DateStamp->ds_Days will not
                 be affected.

       dat_StrTime -   Pointer to a buffer which contains the time in
                 the ASCII format hh:mm:ss.  This will be converted
                 to the ds_Minutes and ds_Ticks portions of the
                 DateStamp.  If this pointer is NULL, ds_Minutes and
                 ds_Ticks will be unchanged.

   RESULT
       success - a zero return indicates that a conversion could
               not be performed. A non-zero return indicates that the
```

```
            DateTime.dat_Stamp variable contains the converted
            values.

   SEE ALSO
        DateStamp(), DateToStr(), <dos/datetime.h>
```

```
dos.library/StrToLong                                   dos.library/StrToLong

   NAME
        StrToLong -- string to long value (decimal) (V36)

   SYNOPSIS
        characters = StrToLong(string,value)
        D0                     D1     D2

        LONG StrToLong(UBYTE *, LONG *)

   FUNCTION
        Converts decimal string into LONG value.  Returns number of characters
        converted.  Skips over leading spaces & tabs (included in count).  If
        no decimal digits are found (after skipping leading spaces & tabs),
        StrToLong returns -1 for characters converted, and puts 0 into value.

   INPUTS
        string - Input string.
        value  - Pointer to long value.  Set to 0 if no digits are converted.

   RESULT
        result - Number of characters converted or -1.
```

```
dos.library/SystemTagList                          dos.library/SystemTagList

   NAME
       SystemTagList -- Have a shell execute a command line (V36)

   SYNOPSIS
       error = SystemTagList(command, tags)
       D0                    D1       D2

       LONG SystemTagList(UBYTE *, struct TagItem *)

       error = System(command, tags)
       D0             D1       D2

       LONG System(UBYTE *, struct TagItem *)

       error = SystemTags(command, Tag1, ...)

       LONG SystemTags(UBYTE *, ULONG, ...)

   FUNCTION
       Similar to Execute(), but does not read commands from the input
       filehandle.  Spawns a Shell process to execute the command, and
       returns the returncode the command produced, or -1 if the command
       could not be run for any reason.  The input and output filehandles
       will not be closed by System, you must close them (if needed) after
       System returns, if you specified them via SYS_INPUT or SYS_OUTPUT.

       By default the new process will use your current Input() and Output()
       filehandles.  Normal Shell command-line parsing will be done
       including redirection on 'command'.  The current directory and path
       will be inherited from your process.  Your path will be used to find
       the command (if no path is specified).

       If used with the SYS_Asynch flag, it WILL close both it's input and
       output filehandles after running the command (even if these were
       your Input() and Output()!)

       Normally uses the boot (ROM) shell, but other shells can be specified
       via SYS_UserShell and SYS_CustomShell.  Normally, you should send
       things written by the user to the UserShell.  The UserShell defaults
       to the same shell as the boot shell.

       The tags are passed through to CreateNewProc() (tags that conflict
       with SystemTagList() will be filtered out).  This allows setting
       things like priority, etc for the new process.

   INPUTS
       command - Program and arguments
       tags    - see <dos/dostags.h>.  Note that both SystemTagList()-
                 specific tags and tags from CreateNewProc() may be passed.

   RESULT
       error   - 0 for success, result from command, or -1.  Note that on
                 error, the caller is responsible for any filehandles or other
                 things passed in via tags.

   SEE ALSO
       Execute(), CreateNewProc(), <dos/dostags.h>, Input(), Output()
```

```
dos.library/UnGetC                                        dos.library/UnGetC

   NAME
       UnGetC -- Makes a char available for reading again. (buffered) (V36)

   SYNOPSIS
       value = UnGetC(fh, character)
       D0             D1      D2

       LONG UnGetC(BPTR, LONG)

   FUNCTION
       Pushes the character specified back into the input buffer.  Every
       time you use a buffered read routine, you can always push back 1
       character.  You may be able to push back more, though it is not
       recommended, since there is no guarantee on how many can be
       pushed back at a given moment.

       Passing -1 for the character will cause the last character read to
       be pushed back.  If the last character read was an EOF, the next
       character read will be an EOF.

       Note: UnGetC can be used to make sure that a filehandle is set up
       as a read filehandle.  This is only of importance if you are writing
       a shell, and must manipulate the filehandle's buffer.

   INPUTS
       fh        - filehandle to use for buffered I/O
       character - character to push back or -1

   RESULT
       value     - character pushed back, or FALSE if the character cannot
                   be pushed back.

   BUGS
       In V36, UnGetC(fh,-1) after an EOF would not cause the next character
       read to be an EOF.  This was fixed for V37.

   SEE ALSO
       FGetC(), FPutC(), Flush()
```

dos.library Page 147

```
dos.library/UnLoadSeg                                   dos.library/UnLoadSeg

    NAME
        UnLoadSeg -- Unload a seglist previously loaded by LoadSeg()

    SYNOPSIS
        success = UnLoadSeg( seglist )
        D0                   D1

        BOOL UnLoadSeg(BPTR)

    FUNCTION
        Unload a seglist loaded by LoadSeg().  'seglist' may be zero.
        Overlaid segments will have all needed cleanup done, including
        closing files.

    INPUTS
        seglist - BCPL pointer to a segment identifier

    RESULTS
        success - returns 0 if a NULL seglist was passed or if it failed
                  to close an overlay file.  NOTE: this function returned
                  a random value before V36!

    SEE ALSO
        LoadSeg(), InternalLoadSeg(), InternalUnLoadSeg()
```

dos.library Page 148

```
dos.library/UnLock                                           dos.library/UnLock

    NAME
        UnLock -- Unlock a directory or file

    SYNOPSIS
        UnLock( lock )
                D1

        void UnLock(BPTR)

    FUNCTION
        The filing system lock (obtained from Lock(), DupLock(), or
        CreateDir()) is removed and deallocated.

    INPUTS
        lock - BCPL pointer to a lock

    NOTE
        passing zero to UnLock() is harmless

    SEE ALSO
        Lock(), DupLock(), ParentOfFH(), DupLockFromFH()
```

```
dos.library/UnLockDosList                             dos.library/UnLockDosList

   NAME
        UnLockDosList -- Unlocks the Dos List (V36)

   SYNOPSIS
        UnLockDosList(flags)
                      D1

        void UnLockDosList(ULONG)

   FUNCTION
        Unlocks the access on the Dos Device List.  You MUST pass the same
        flags you used to lock the list.

   INPUTS
        flags - MUST be the same flags passed to (Attempt)LockDosList()

   SEE ALSO
        AttemptLockDosList(), LockDosList(), Permit()
```

```
dos.library/UnLockRecord                                   dos.library/UnLockRecord

   NAME
        UnLockRecord -- Unlock a record (V36)

   SYNOPSIS
        success = UnLockRecord(fh,offset,length)
        D0                     D1   D2     D3

        BOOL UnLockRecord(BPTR,ULONG,ULONG)

   FUNCTION
        This releases the specified lock on a file.  Note that you must use
        the same filehandle you used to lock the record, and offset and length
        must be the same values used to lock it.  Every LockRecord() call must
        be balanced with an UnLockRecord() call.

   INPUTS
        fh      - File handle of locked file
        offset  - Record start position
        length  - Length of record in bytes

   RESULT
        success - Success or failure.

   BUGS
        See LockRecord()

   SEE ALSO
        LockRecords(), LockRecord(), UnLockRecords()
```

dos.library Page 151

```
dos.library/UnLockRecords                            dos.library/UnLockRecords

   NAME
        UnLockRecords -- Unlock a list of records (V36)

   SYNOPSIS
        success = UnLockRecords(record_array)
        D0                      D1

        BOOL UnLockRecords(struct RecordLock *)

   FUNCTION
        This releases an array of record locks obtained using LockRecords.
        You should NOT modify the record_array while you have the records
        locked.  Every LockRecords() call must be balanced with an
        UnLockRecords() call.

   INPUTS
        record_array - List of records to be unlocked

   RESULT
        success      - Success or failure.

   BUGS
        See LockRecord()

   SEE ALSO
        LockRecords(), LockRecord(), UnLockRecord()
```

dos.library Page 152

```
dos.library/VFPrintf                                      dos.library/VFPrintf

   NAME
        VFPrintf -- format and print a string to a file (buffered) (V36)

   SYNOPSIS
        count = VFPrintf(fh, fmt, argv)
        D0                D1  D2   D3

        LONG VFPrintf(BPTR, char *, LONG *)

        count = FPrintf(fh, fmt, ...)

        LONG FPrintf(BPTR, char *, ...)

   FUNCTION
        Writes the formatted string and values to the given file.  This
        routine is assumed to handle all internal buffering so that the
        formatting string and resultant formatted values can be arbitrarily
        long.  Any secondary error code is returned in IoErr().  This routine
        is buffered.

   INPUTS
        fh   - Filehandle to write to
        fmt  - RawDoFmt() style formatting string
        argv - Pointer to array of formatting values

   RESULT
        count - Number of bytes written or -1 (EOF) for an error

   SEE ALSO
        VPrintf(), VFWritef(), RawDoFmt(), FPutC()
```

```
dos.library/VFWritef                                  dos.library/VFWritef

   NAME
        VFWritef - write a BCPL formatted string to a file (buffered) (V36)

   SYNOPSIS
        count = VFWritef(fh, fmt, argv)
        D0               D1  D2    D3

        LONG VFWritef(BPTR, char *, LONG *)

        count = FWritef(fh, fmt, ...)

        LONG FWritef(BPTR, char *, ...)

   FUNCTION
        Writes the formatted string and values to the default output.  This
        routine is assumed to handle all internal buffering so that the
        formatting string and resultant formatted values can be arbitrarily
        long.  The formats are in BCPL form.  This routine is buffered.

        Supported formats are:  (Note x is in base 36!)
                %S  - string (CSTR)
                %Tx - writes a left-justified string in a field at least
                      x bytes long.
                %C  - writes a single character
                %Ox - writes a number in octal, maximum x characters wide
                %Xx - writes a number in hex, maximum x characters wide
                %Ix - writes a number in decimal, maximum x characters wide
                %N  - writes a number in decimal, any length
                %Ux - writes an unsigned number, maximum x characters wide
                %$  - ignore parameter

   INPUTS
        fmt   - BCPL style formatting string
        argv  - Pointer to array of formatting values

   RESULT
        count - Number of bytes written or -1 for error

   SEE ALSO
        VFPrintf(), VFPrintf(), FPutC()
```

```
dos.library/VPrintf                                    dos.library/VPrintf

   NAME
        VPrintf -- format and print string (buffered) (V36)

   SYNOPSIS
        count = VPrintf(fmt, argv)
          D0            D1   D2

        LONG VPrintf(char *, LONG *)

        count = Printf(fmt, ...)

        LONG Printf(char *, ...)

   FUNCTION
        Writes the formatted string and values to Output().  This routine is
        assumed to handle all internal buffering so that the formatting string
        and resultant formatted values can be arbitrarily long.  Any secondary
        error code is returned in IoErr().  This routine is buffered.

        Note: RawDoFmt assumes 16 bit ints, so you will usually need 'l's in
        your formats (ex: %ld versus %d).

   INPUTS
        fmt   - exec.library RawDoFmt() style formatting string
        argv  - Pointer to array of formatting values

   RESULT
        count - Number of bytes written or -1 (EOF) for an error

   SEE ALSO
        VFPrintf(), VFWritef(), RawDoFmt(), FPutC()
```

dos.library Page 155

```
dos.library/WaitForChar                            dos.library/WaitForChar

   NAME
       WaitForChar -- Determine if chars arrive within a time limit

   SYNOPSIS
       status = WaitForChar( file, timeout )
       D0                    D1    D2

       BOOL WaitForChar(BPTR, LONG)

   FUNCTION
       If a character is available to be read from 'file' within a the
       time (in microseconds) indicated by 'timeout', WaitForChar()
       returns -1 (TRUE). If a character is available, you can use Read()
       to read it.  Note that WaitForChar() is only valid when the I/O
       stream is connected to a virtual terminal device. If a character is
       not available within 'timeout', a 0 (FALSE) is returned.

   BUGS
       Due to a bug in the timer.device in V1.2/V1.3, specifying a timeout
       of zero for WaitForChar() can cause the unreliable timer & floppy
       disk operation.

   INPUTS
       file - BCPL pointer to a file handle
       timeout - integer

   RESULTS
       status - boolean

   SEE ALSO
       Read(), FGetC()
```

dos.library Page 156

```
dos.library/WaitPkt                                            dos.library/WaitPkt

   NAME
       WaitPkt -- Waits for a packet to arrive at your pr_MsgPort (V36)

   SYNOPSIS
       packet = WaitPkt()
       D0

       struct DosPacket *WaitPkt(void);

   FUNCTION
       Waits for a packet to arrive at your pr_MsgPort.  If anyone has
       installed a packet wait function in pr_PktWait, it will be called.
       The message will be automatically GetMsg()ed so that it is no longer
       on the port.  It assumes the message is a dos packet.  It is NOT
       guaranteed to clear the signal for the port.

   RESULT
       packet - the packet that arrived at the port (from ln_Name of message).

   SEE ALSO
       SendPkt(), DoPkt(), AbortPkt()
```

dos.library Page 157

```
dos.library/Write                                           dos.library/Write

   NAME
       Write -- Write bytes of data to a file

   SYNOPSIS
       returnedLength =  Write( file, buffer, length )
       D0                       D1    D2      D3

       LONG Write (BPTR, void *, LONG)

   FUNCTION
       Write() writes bytes of data to the opened file 'file'. 'length'
       indicates the length of data to be transferred; 'buffer' is a
       pointer to the buffer. The value returned is the length of
       information actually written. So, when 'length' is greater than
       zero, the value of 'length' is the number of characters written.
       Errors are indicated by a value of -1.

       Note: this is an unbuffered routine (the request is passed directly
       to the filesystem.)  Buffered I/O is more efficient for small
       reads and writes; see FPutC().

   INPUTS
       file - BCPL pointer to a file handle
       buffer - pointer to the buffer
       length - integer

   RESULTS
       returnedLength - integer

   SEE ALSO
       Read(), Seek(), Open(), Close(), FPutC
```

dos.library Page 158

```
dos.library/WriteChars                                 dos.library/WriteChars

   NAME
       WriteChars -- Writes bytes to the the default output (buffered) (V36)

   SYNOPSIS
       count = WriteChars(buf, buflen)
       D0                 D1

       LONG PutStr(UBYTE *, LONG)

   FUNCTION
       This routine writes a number of bytes to the default output.  The
       length is returned.  This routine is buffered.

   INPUTS
       buf    - buffer of characters to write
       buflen - number of characters to write

   RESULT
       count - Number of bytes written.  -1 (EOF) indicates an error

   SEE ALSO
       FPuts(), FPutC(), FWrite(), PutStr()
```

exec.library

Page 1

TABLE OF CONTENTS

exec.library/AbortIO
exec.library/AddDevice
exec.library/AddHead
exec.library/AddIntServer
exec.library/AddLibrary
exec.library/AddMemList
exec.library/AddPort
exec.library/AddResource
exec.library/AddSemaphore
exec.library/AddTail
exec.library/AddTask
exec.library/Alert
exec.library/AllocAbs
exec.library/Allocate
exec.library/AllocEntry
exec.library/AllocMem
exec.library/AllocSignal
exec.library/AllocTrap
exec.library/AllocVec
exec.library/AttemptSemaphore
exec.library/AvailMem
exec.library/CacheClearE
exec.library/CacheClearU
exec.library/CacheControl
exec.library/Cause
exec.library/CheckIO
exec.library/CloseDevice
exec.library/CloseLibrary
exec.library/ColdReboot
exec.library/CopyMem
exec.library/CopyMemQuick
exec.library/CreateIORequest
exec.library/CreateMsgPort
exec.library/Deallocate
exec.library/Debug
exec.library/DeleteIORequest
exec.library/DeleteMsgPort
exec.library/Disable
exec.library/DoIO
exec.library/Enable
exec.library/Enqueue
exec.library/FindName
exec.library/FindPort
exec.library/FindResident
exec.library/FindSemaphore
exec.library/FindTask
exec.library/Forbid
exec.library/FreeEntry
exec.library/FreeMem
exec.library/FreeSignal
exec.library/FreeTrap
exec.library/FreeVec
exec.library/GetCC
exec.library/GetMsg
exec.library/InitCode
exec.library/InitResident
exec.library/InitSemaphore
exec.library/InitStruct
exec.library/Insert
exec.library/MakeFunctions
exec.library/MakeLibrary
exec.library/ObtainSemaphore
exec.library/ObtainSemaphoreList
exec.library/ObtainSemaphoreShared

exec.library

Page 2

exec.library/OldOpenLibrary
exec.library/OpenDevice
exec.library/OpenLibrary
exec.library/OpenResource
exec.library/Permit
exec.library/Procure
exec.library/PutMsg
exec.library/RawDoFmt
exec.library/ReleaseSemaphore
exec.library/ReleaseSemaphoreList
exec.library/RemDevice
exec.library/RemHead
exec.library/RemIntServer
exec.library/RemLibrary
exec.library/Remove
exec.library/RemPort
exec.library/RemResource
exec.library/RemSemaphore
exec.library/RemTail
exec.library/RemTask
exec.library/ReplyMsg
exec.library/SendIO
exec.library/SetExcept
exec.library/SetFunction
exec.library/SetIntVector
exec.library/SetSignal
exec.library/SetSR
exec.library/SetTaskPri
exec.library/Signal
exec.library/SumKickData
exec.library/SumLibrary
exec.library/SuperState
exec.library/Supervisor
exec.library/TypeOfMem
exec.library/UserState
exec.library/Vacate
exec.library/Wait
exec.library/WaitIO
exec.library/WaitPort

exec.library

Page 3

```
exec.library/AbortIO                                       exec.library/AbortIO

   NAME
       AbortIO - attempt to abort an in-progress I/O request

   SYNOPSIS
       AbortIO(iORequest)
               A1

       VOID AbortIO(struct IORequest *);

   FUNCTION
       Ask a device to abort a previously started IORequest.  This is done
       by calling the device's ABORTIO vector, with your given IORequest.


       AbortIO is a command the device that may or may not grant.  If
       successful, the device will stop processing the IORequest, and
       reply to it earlier than it would otherwise have done.

   NOTE
       AbortIO() does NOT Remove() the IORequest from your ReplyPort, OR
       wait for it to complete.  After an AbortIO() you must wait normally
       for the reply message before actually reusing the request.

       If a request has already completed when AbortIO() is called, no
       action is taken.

   EXAMPLE
           AbortIO(timer_request);
           WaitIO(timer_request);
           /* Message is free to be reused */

   INPUTS
       iORequest - pointer to an I/O request block (must have been used
               at least once.  May be active or finished).

   SEE ALSO
       WaitIO, DoIO, SendIO, CheckIO
```

exec.library

Page 4

```
exec.library/AddDevice                                   exec.library/AddDevice

   NAME
       AddDevice -- add a device to the system

   SYNOPSIS
       AddDevice(device)
                 A1

       void AddDevice(struct Device *);

   FUNCTION
       This function adds a new device to the system device list, making
       it available to other programs.  The device must be ready to be
       opened at this time.

   INPUTS
       device - pointer to a properly initialized device node

   SEE ALSO
       RemDevice, OpenDevice, CloseDevice, MakeLibrary
```

exec.library Page 5

```
exec.library/AddHead                                    exec.library/AddHead

   NAME
        AddHead -- insert node at the head of a list

   SYNOPSIS
        AddHead(list, node)
                A0    A1

        void AddHead(struct List *, struct Node *)

   FUNCTION
        Add a node to the head of a doubly linked list. Assembly
        programmers may prefer to use the ADDHEAD macro from
        "exec/lists.i".

   WARNING
        This function does not arbitrate for access to the list.  The
        calling task must be the owner of the involved list.

   INPUTS
        list - a pointer to the target list header
        node - the node to insert at head

   SEE ALSO
        AddTail, Enqueue, Insert, Remove, RemHead, RemTail
```

exec.library Page 6

```
exec.library/AddIntServer                          exec.library/AddIntServer

   NAME
        AddIntServer -- add an interrupt server to a system server chain

   SYNOPSIS
        AddIntServer(intNum, interrupt)
                     D0-0:4  A1

        void AddIntServer(ULONG, struct Interrupt *);

   FUNCTION
        This function adds a new interrupt server to a given server chain.
        The node is located on the chain in a priority dependent position.
        If this is the first server on a particular chain, interrupts will
        be enabled for that chain.

        Each link in the chain will be called in priority order until the
        chain ends or one of the servers returns with the 68000's Z condition
        code clear (indicating non-zero).  Servers on the chain should return
        with the Z flag clear if the interrupt was specifically for that
        server, and no one else.  VERTB servers should always return Z set.
        (Take care with High Level Language servers, the language may not
        have a mechanism for reliably setting the Z flag on exit).

        Servers are called with the following register conventions:

            D0 - scratch
            D1 - scratch

            A0 - scratch
            A1 - server is_Data pointer (scratch)

            A5 - jump vector register (scratch)
            A6 - scratch

            all other registers must be preserved

   INPUTS
        intNum - the Portia interrupt bit number (0 through 14). Processor
                 level seven interrupts (NMI) are encoded as intNum 15.
                 The PORTS, COPER, VERTB, EXTER and NMI interrupts are
                 set up as server chains.
        interrupt - pointer to an Interrupt structure.
                 By convention, the LN_NAME of the interrupt structure must
                 point a descriptive string so that other users may
                 identify who currently has control of the interrupt.

   WARNING
        Some compilers or assemblers may optimize code in unexpected ways,
        affecting the conditions codes returned from the function.  Watch
        out for a "MOVEM" instruction (which does not affect the condition
        codes) turning into "MOVE" (which does).

   BUGS
        The graphics library's VBLANK server currently assumes that
        address register A0 will contain a pointer to the custom chips.  If
        you add a server at a priority of 10 or greater, you must
        compensate for this by providing the expected value ($DFF000).

   SEE ALSO
        RemIntServer, SetIntVector, hardware/intbits.i,exec/interrupts.i
```

exec.library Page 7

```
exec.library/AddLibrary                                exec.library/AddLibrary

   NAME
        AddLibrary -- add a library to the system

   SYNOPSIS
        AddLibrary(library)
                   A1

        void AddLibrary(struct Library *);

   FUNCTION
        This function adds a new library to the system, making it available
        to other programs.  The library should be ready to be opened at
        this time.  It will be added to the system library name list, and
        the checksum on the library entries will be calculated.

   INPUTS
        library - pointer to a properly initialized library structure

   SEE ALSO
        RemLibrary, CloseLibrary, OpenLibrary, MakeLibrary
```

exec.library Page 8

```
exec.library/AddMemList                                exec.library/AddMemList

   NAME
        AddMemList - add memory to the system free pool

   SYNOPSIS
        AddMemList( size, attributes, pri, base, name )
                    D0    D1          D2   A0    A1

        void AddMemList(ULONG, ULONG, LONG, APTR, STRPTR);

   FUNCTION
        Add a new region of memory to the system free pool.  The first few
        bytes will be used to hold the MemHeader structure.  The remainder
        will be made available to the rest of the world.

   INPUTS
        size - the size (in bytes) of the memory area
        attributes - the attributes word that the memory pool will have
        pri  - the priority for this memory.  CHIP memory has a pri of -10,
               16 bit expansion memory has a priority of 0.  The higher the
               priority, the closer to the head of the memory list it will
               be placed.
        base - the base of the new memory area
        name - the name that will be used in the memory header, or NULL
               if no name is to be provided.  This name is not copied, so it
               must remain valid for as long as the memory header is in the
               system.

   SEE ALSO
        AllocMem, exec/memory.h
```

exec.library Page 9

```
exec.library/AddPort                                      exec.library/AddPort

   NAME
       AddPort -- add a public message port to the system

   SYNOPSIS
       AddPort(port)
               A1

       void AddPort(struct MsgPort *);

   FUNCTION
       This function attaches a message port structure to the system's
       public message port list, where it can be found by the FindPort()
       function.  The name and priority fields of the port structure must
       be initialized prior to calling this function.  If the user does
       not require the priority field, it should be initialized to zero.

       Only ports that will be searched for with FindPort() need to
       be added to the system list.  In addition, adding ports is often
       useful during debugging.  If the port will be searched for,
       the priority field should be at least 1 (to avoid the large number
       of inactive ports at priority zero).  If the port will be searched
       for often, set the proritiry in the 50-100 range (so it will be
       before other less used ports).

       Once a port has been added to the naming list, you must be careful
       to remove the port from the list (via RemPort) before deallocating
       its memory.

   NOTE
       A point of confusion is that clearing a MsgPort structure to all
       zeros is not enough to prepare it for use.  As mentioned in the
       Exec chapter of the ROM Kernel Manual, the List for the MsgPort
       must be initialized.  This is automatically handled by AddPort(),
       and amiga.lib/CreatePort.  This initialization can be done manually
       with amiga.lib/NewList or the assembly NEWLIST macro.

       Do not AddPort an active port.

   INPUTS
       port - pointer to a message port

   SEE ALSO
       RemPort, FindPort, amiga.lib/CreatePort, amiga.lib/NewList
```

exec.library Page 10

```
exec.library/AddResource                              exec.library/AddResource

   NAME
       AddResource -- add a resource to the system

   SYNOPSIS
       AddResource(resource)
                   A1

       void AddResource(APTR);

   FUNCTION
       This function adds a new resource to the system and makes it
       available to other users.  The resource must be ready to be called
       at this time.

       Resources currently have no system-imposed structure, however they
       must start with a standard named node (LN_SIZE), and should with
       a standard Library node (LIB_SIZE).

   INPUTS
       resource - pointer an initialized resource node

   SEE ALSO
       RemResource, OpenResource, MakeLibrary
```

```
exec.library/AddSemaphore                         exec.library/AddSemaphore

   NAME
        AddSemaphore -- initialize then add a signal semaphore to the system

   SYNOPSIS
        AddSemaphore(signalSemaphore)
                     A1

        void AddSemaphore(struct SignalSemaphore *);

   FUNCTION
        This function attaches a signal semaphore structure to the system's
        public signal semaphore list.  The name and priority fields of the
        semaphore structure must be initialized prior to calling this
        function.  If you do not want to let others rendezvous with this
        semaphore, use InitSemaphore() instead.

        If a semaphore has been added to the naming list, you must be
        careful to remove the semaphore from the list (via RemSemaphore)
        before deallocating its memory.

        Semaphores that are linked together in an allocation list (which
        ObtainSemaphoreList() would use) may not be added to the system
        naming list, because the facilities use the link field of the
        signal semaphore in incompatible ways

   INPUTS
       signalSemaphore -- an signal semaphore structure

   BUGS
        Does not work in Exec <V36.  Instead use this code:

            #include <exec/execbase.h>
            #include <exec/nodes.h>
            extern struct ExecBase *SysBase;
                ...
            void LocalAddSemaphore(s)
            struct SignalSemaphore *s;
            {
                s->ss_Link.ln_Type=NT_SIGNALSEM;
                InitSemaphore(s);
                Forbid();
                Enqueue(&SysBase->SemaphoreList,s);
                Permit();
            }

   SEE ALSO
        RemSemaphore, FindSemaphore, InitSemaphore
```

```
exec.library/AddTail                                 exec.library/AddTail

   NAME
        AddTail -- append node to tail of a list

   SYNOPSIS
        AddTail(list, node)
                A0    A1

        void AddTail(struct List *, struct Node *);

   FUNCTION
        Add a node to the tail of a doubly linked list.  Assembly
        programmers may prefer to use the ADDTAIL macro from
        "exec/lists.i".

   WARNING
        This function does not arbitrate for access to the list.  The
        calling task must be the owner of the involved list.

   INPUTS
        list - a pointer to the target list header
        node - a pointer to the node to insert at tail of the list

   SEE ALSO
        AddHead, Enqueue, Insert, Remove, RemHead, RemTail
```

```
exec.library/AddTask                                    exec.library/AddTask

   NAME
       AddTask -- add a task to the system

   SYNOPSIS
       AddTask(task, initialPC, finalPC)
               A1    A2         A3

       APTR AddTask(struct Task *, APTR, APTR);

   FUNCTION
       Add a task to the system.  A reschedule will be run; the task with
       the highest priority in the system will start to execute (this may
       or may not be the new task).

       Certain fields of the task control block must be initialized and a
       stack allocated prior to calling this function.  The absolute
       smallest stack that is allowable is something in the range of 100
       bytes, but in general the stack size is dependent on what
       subsystems are called. In general 256 bytes is sufficient if only
       Exec is called, and 4K will do if anything in the system is called.
       DO NOT UNDERESTIMATE.  If you use a stack sniffing utility,
       leave a healthy pad above the minimum value.  The system guarantees
       that it's stack operations will leave the stack longword aligned.

       This function will temporarily use space from the new task's stack
       for the task's initial set of registers.  This space is allocated
       starting at the SPREG location specified in the task control block
       (not from SPUPPER).  This means that a task's stack may contain
       static data put there prior to its execution.  This is useful for
       providing initialized global variables or some tasks may want to
       use this space for passing the task its initial arguments.

       A task's initial registers are set to zero (except the PC).

       The TC_MEMENTRY field of the task structure may be extended by
       the user to hold additional MemLists (as returned by AllocEntry()).
       These will be automatically be deallocated at RemTask() time.
       If the code you have used to start the task has already added
       something to the MEMENTRY list, simply use AddHead to add your
       new MemLists in.  If no initialization has been done, a NewList will
       need to be performed.

   NOTE
       AddTask clears out TC_FLAGS.

   INPUTS
       task - pointer to the task control block (TCB)
       initialPC - the initial entry point's address
       finalPC - the finalization code entry point's address.  If zero,
                 the system will use a general finalizer. This pointer is
                 placed on the stack as if it were the outermost return
                 address.

   RESULTS
       For V36, AddTask returns either a NULL or the address of the new
       task.  Old code need not check this.

   WARNING
       Tasks are a low-level building block, and are unable to call
       dos.library, or any system function that might call dos.library.
       See the AmigaDOS CreateProc() for information on Processes.

   SEE ALSO
       RemTask, FindTask, amiga.lib/CreateTask, dos/CreateProc,
```

```
       amiga.lib/NewList
```

```
exec.library/Alert                                          exec.library/Alert

   NAME
        Alert -- alert the user of an error

   SYNOPSIS
        Alert(alertNum)
              D7

        void Alert(ULONG);

   FUNCTION
        Alerts the user of a serious system problem.  This function will
        bring the system to a grinding halt, and do whatever is necessary
        to present the user with a message stating what happened.
        Interrupts are disabled, and an attempt to post the alert is made.
        If that fails, the system is reset.  When the system comes up
        again, Exec notices the cause of the failure and tries again to
        post the alert.

        If the Alert is a recoverable type, this call MAY return.

        This call may be made at any time, including interrupts.

   POST-MORTEM DIAGNOSIS
        There are several options for determining the cause of a crash.
        Descriptions of each alert number can be found in the "alerts.h"
        include file.  Low numbers not mentioned in the include file
        represent 68000 exceptions, see a 68000 manual for details.  The
        most common numbers are:
                        $00000003 - Address Error
                        $00000004 - Illegal Instruction

        A remote terminal can be attached to the Amiga's first built-in
        serial port.  Set the communication parameters to 9600 baud, 8 bits,
        no parity.  Before resetting the machine, the Alert function will
        blink the power LED 10 times.  While the power indicator is flashing,
        pressing DELETE on the remote terminal will invoke the ROMWack
        debugger.

        For Alerts caused by a 68000 exception, all registers are copied
        to a magic low memory location (currently 16 longwords at $180).

   INPUT
        alertNum   - a number indicating the particular alert.  -1 is
                not a valid input.

   NOTE
        Much more needs to be said about this function and its implications.

   SEE ALSO
        exec/alerts.h
```

```
exec.library/AllocAbs                                    exec.library/AllocAbs

   NAME
        AllocAbs -- allocate at a given location

   SYNOPSIS
        memoryBlock = AllocAbs(byteSize, location)
        D0                     D0        A1

        void *AllocAbs(ULONG, APTR);

   FUNCTION
        This function attempts to allocate memory at a given absolute
        memory location.  Often this is used by boot-surviving entities
        such as recoverable ram-disks.  If the memory is already being
        used, or if there is not enough memory to satisfy the request,
        AllocAbs will return NULL.

        This block may not be exactly the same as the requested block
        because of rounding, but if the return value is non-zero, the block
        is guaranteed to contain the requested range.

   INPUTS
        byteSize - the size of the desired block in bytes
                   This number is rounded up to the next larger
                   block size for the actual allocation.
        location - the address where the memory MUST be.


   RESULT
        memoryBlock - a pointer to the newly allocated memory block, or
                      NULL if failed.

   NOTE
        If the free list is corrupt, the system will panic with alert
        AN_MemCorrupt, $01000005.

        The 8 bytes past the end of an AllocAbs will be changed by Exec
        relinking the next block of memory.  Generally you can't trust
        the first 8 bytes of anything you AllocAbs.

   SEE ALSO
        AllocMem, FreeMem
```

```
exec.library/Allocate                                        exec.library/Allocate

   NAME
        Allocate - allocate a block of memory

   SYNOPSIS
        memoryBlock=Allocate(memHeader, byteSize)
        D0                   A0         D0

        void *Allocate(struct MemHeader *, ULONG);

   FUNCTION
        This function is used to allocate blocks of memory from a given
        private free memory pool (as specified by a MemHeader and its
        memory chunk list).  Allocate will return the first free block that
        is greater than or equal to the requested size.

        All blocks, whether free or allocated, will be block aligned;
        hence, all allocation sizes are rounded up to the next block even
        value (e.g. the minimum allocation resolution is currently 8
        bytes.  A request for 8 bytes will use up exactly 8 bytes.  A
        request for 7 bytes will also use up exactly 8 bytes.).

        This function can be used to manage an application's internal data
        memory.  Note that no arbitration of the MemHeader and associated
        free chunk list is done.  You must be the owner before calling
        Allocate.

   INPUTS
        memHeader - points to the local memory list header.
        byteSize - the size of the desired block in bytes.

   RESULT
        memoryBlock - a pointer to the just allocated free block.
               If there are no free regions large enough to satisfy the
               request, return zero.

   EXAMPLE
        #include <exec/types.h>
        #include <exec/memory.h>
        void *AllocMem();
        #define BLOCKSIZE 4096L /* Or whatever you want */

        void main()
        {
        struct MemHeader *mh;
        struct MemChunk  *mc;
        APTR   block1;
        APTR   block2;

            /* Get the MemHeader needed to keep track of our new block */
            mh = (struct MemHeader *)
                 AllocMem((long)sizeof(struct MemHeader), MEMF_CLEAR );
            if( !mh )
                exit(10);

            /* Get the actual block the above MemHeader will manage */
            mc = (struct MemChunk *)AllocMem( BLOCKSIZE, 0L );
            if( !mc )
                {
                FreeMem( mh, (long)sizeof(struct MemHeader) ); exit(10);
                }

            mh->mh_Node.ln_Type = NT_MEMORY;
            mh->mh_Node.ln_Name = "myname";
            mh->mh_First = mc;
```

```
            mh->mh_Lower = (APTR) mc;
            mh->mh_Upper = (APTR) ( BLOCKSIZE + (ULONG) mc );
            mh->mh_Free  = BLOCKSIZE;

            /* Set up first chunk in the freelist */
            mc->mc_Next  = NULL;
            mc->mc_Bytes = BLOCKSIZE;

            block1 = (APTR) Allocate( mh, 20L );
            block2 = (APTR) Allocate( mh, 314L );
            printf("mh=$%lx mc=$%lx\n",mh,mc);
            printf("Block1=$%lx, Block2=$%lx\n",block1,block2);

            FreeMem( mh, (long)sizeof(struct MemHeader) );
            FreeMem( mc, BLOCKSIZE );
        }

   NOTE
        If the free list is corrupt, the system will panic with alert
        AN_MemCorrupt, $01000005.

   SEE ALSO
        Deallocate, exec/memory.h
```

```
exec.library/AllocEntry                              exec.library/AllocEntry

   NAME
       AllocEntry -- allocate many regions of memory

   SYNOPSIS
       memList = AllocEntry(memList)
       D0                   A0

       struct MemList *AllocEntry(struct MemList *);

   FUNCTION
       This function takes a memList structure and allocates enough memory
       to hold the required memory as well as a MemList structure to keep
       track of it.

       These MemList structures may be linked together in a task control
       block to keep track of the total memory usage of this task. (See
       the description of TC_MEMENTRY under RemTask).

   INPUTS
       memList -- A MemList structure filled in with MemEntry structures.

   RESULTS
       memList -- A different MemList filled in with the actual memory
           allocated in the me_Addr field, and their sizes in me_Length.
           If enough memory cannot be obtained, then the requirements of
           the allocation that failed is returned and bit 31 is set.

   EXAMPLES
       The user wants five regions of 2, 4, 8, 16, and 32 bytes in size
       with requirements of MEMF_CLEAR, MEMF_PUBLIC, MEMF_CHIP!MEMF_CLEAR,
       MEMF_CLEAR, and MEMF_PUBLIC!MEMF_CLEAR respectively.  The
       following code fragment would do that:

           MemListDecl:
               DS.B    LN_SIZE             * reserve space for list node
               DC.W    5                   * number of entries
               DC.L    MEMF_CLEAR                  * entry #0
               DC.L    2
               DC.L    MEMF_PUBLIC                 * entry #1
               DC.L    4
               DC.L    MEMF_CHIP!MEMF_CLEAR        * entry #2
               DC.L    8
               DC.L    MEMF_CLEAR                  * entry #3
               DC.L    16
               DC.L    MEMF_PUBLIC!MEMF_CLEAR      * entry #4
               DC.L    32

           start:
               LEA.L   MemListDecl(PC),A0
               JSR     _LVOAllocEntry(a6)
               BCLR.L  #31,D0
               BEQ.S   success

               ------- Type of memory that we failed on is in D0

   BUGS
       If any one of the allocations fails, this function fails to back
       out fully.  This is fixed by the "SetPatch" program on V1.3
       Workbench disks.

   SEE ALSO
       exec/memory.h
```

```
exec.library/AllocMem                                  exec.library/AllocMem

   NAME
       AllocMem -- allocate memory given certain requirements

   SYNOPSIS
       memoryBlock = AllocMem(byteSize, attributes)
       D0                     D0        D1

       void *AllocMem(ULONG, ULONG);

   FUNCTION
       This is the memory allocator to be used by system code and
       applications.  It provides a means of specifying that the allocation
       should be made in a memory area accessible to the chips, or
       accessible to shared system code.

       Memory is allocated based on requirements and options.  Any
       "requirement" must be met by a memory allocation, any "option" will
       be applied to the block regardless.  AllocMem will try all memory
       spaces until one is found with the proper requirements and room for
       the memory request.

   INPUTS
       byteSize - the size of the desired block in bytes.  (The operating
               system will automatically round this number to a multiple of
               the system memory chunk size)

       attributes -
           requirements

               If no flags are set, the system will return the best
               available memory block.  For expanded systems, the fast
               memory pool is searched first.

               MEMF_CHIP:      If the requested memory will be used by
                               the Amiga custom chips, this flag *must*
                               be set.

                               Only certain parts of memory are reachable
                               by the special chip sets' DMA circuitry.
                               Chip DMA includes screen memory, images that
                               are blitted, audio data, copper lists, sprites
                               and trackdisk.device buffers.


               MEMF_FAST:      This is non-chip memory.  If no flag is set
                               MEMF_FAST is taken as the default.

                               DO NOT SPECIFY MEMF_FAST unless you know
                               exactly what you are doing!  If MEMF_FAST is
                               set, AllocMem() will fail on machines that
                               only have chip memory!  This flag may not
                               be set when MEMF_CHIP is set.


               MEMF_PUBLIC:    Memory that must not be mapped, swapped,
                               or otherwise made non-addressable. ALL
                               MEMORY THAT IS REFERENCED VIA INTERRUPTS
                               AND/OR BY OTHER TASKS MUST BE EITHER PUBLIC
                               OR LOCKED INTO MEMORY! This includes both
                               code and data.

           options

               MEMF_CLEAR:     The memory will be initialized to all
```

```
                        zeros.

   RESULT
        memoryBlock - a pointer to the newly allocated memory block.
                If there are no free memory regions large enough to satisfy
                the request, zero will be returned.  The pointer must be
                checked for zero before the memory block may be used!

   WARNING
        The result of any memory allocation MUST be checked, and a viable
        error handling path taken.  ANY allocation may fail if memory has
        been filled.

   EXAMPLES
        AllocMem(64,0L)         - Allocate the best available memory
        AllocMem(25,MEMF_CLEAR) - Allocate the best available memory, and
                                  clear it before returning.
        AllocMem(128,MEMF_CHIP) - Allocate chip memory
        AllocMem(128,MEMF_CHIP|MEMF_CLEAR) - Allocate cleared chip memory
        AllocMem(821,MEMF_CHIP|MEMF_PUBLIC|MEMF_CLEAR) - Allocate cleared,
                public, chip memory.

   NOTE
        If the free list is corrupt, the system will panic with alert
        AN_MemCorrupt, $01000005.

        This function may not be called from interrupts.

        A DOS process will have its pr_Result2 field set to
        ERROR_NO_FREE_STORE if the memory allocation fails.

   SEE ALSO
        FreeMem
```

```
exec.library/AllocSignal                             exec.library/AllocSignal

   NAME
        AllocSignal -- allocate a signal bit

   SYNOPSIS
        signalNum = AllocSignal(signalNum)
        D0                      D0

        BYTE AllocSignal(BYTE);

   FUNCTION
        Allocate a signal bit from the current tasks' pool.  Either a
        particular bit, or the next free bit may be allocated.  The signal
        associated with the bit will be properly initialized (cleared).  At
        least 16 user signals are available per task.  Signals should be
        deallocated before the task exits.

        If the signal is already in use (or no free signals are available)
        a -1 is returned.

        Allocated signals are only valid for use with the task that
        allocated them.


   WARNING
        Signals may not be allocated or freed from exception handling code.

   INPUTS
        signalNum - the desired signal number {of 0..31} or -1 for no
                    preference.

   RESULTS
        signalNum - the signal bit number allocated {0..31}. If no signals
                    are available, this function returns -1.

   SEE ALSO
        FreeSignal
```

```
exec.library/AllocTrap                                   exec.library/AllocTrap

   NAME
        AllocTrap -- allocate a processor trap vector

   SYNOPSIS
        trapNum = AllocTrap(trapNum)
        D0                  D0

        LONG AllocTrap(LONG);

   FUNCTION
        Allocate a trap number from the current task's pool.  These trap
        numbers are those associated with the 68000 TRAP type instructions.
        Either a particular number, or the next free number may be
        allocated.

        If the trap is already in use (or no free traps are available) a -1
        is returned.

        This function only affects the currently running task.

        Traps are sent to the trap handler pointed at by tc_TrapCode.
        Unless changed by user code, this points to a standard trap
        handler.  The stack frame of the exception handler will be:

                0(SP) = Exception vector number.  This will be in the
                        range of 32 to 47 (corresponding to the
                        Trap #1...Trap #15 instructions).
                4(SP) = 68000/68010/68020/68030, etc. exception frame

        tc_TrapData is not used.


   WARNING
        Traps may not be allocated or freed from exception handling code.
        You are not allowed to write to the exception table yourself.  In
        fact, on some machines you will have trouble finding it - the VBR
        register may be used to remap its location.

   INPUTS
        trapNum - the desired trap number {of 0..15} or -1
                  for no preference.

   RESULTS
        trapNum - the trap number allocated {of 0..15}.  If no traps are
                  available, this function returns -1.  Instructions of the
                  form "Trap #trapNum" will be sent to the task's trap
                  handler.

   SEE ALSO
        FreeTrap
```

```
exec.library/AllocVec                                     exec.library/AllocVec

   NAME
        AllocVec -- allocate memory and keep track of the size  (V36)

   SYNOPSIS
        memoryBlock = AllocVec(byteSize, attributes)
        D0                     D0        D1

        void *AllocVec(ULONG, ULONG);

   FUNCTION
        This function works identically to AllocMem(), but tracks the size
        of the allocation.

        See the AllocMem() documentation for details.

   WARNING
        The result of any memory allocation MUST be checked, and a viable
        error handling path taken.  ANY allocation may fail if memory has
        been filled.

   SEE ALSO
        FreeVec, AllocMem
```

exec.library Page 25

```
exec.library/AttemptSemaphore                    exec.library/AttemptSemaphore

   NAME
        AttemptSemaphore -- try to obtain without blocking

   SYNOPSIS
        success = AttemptSemaphore(signalSemaphore)
        D0                         A0

        LONG AttemptSemaphore(struct SignalSemaphore *);

   FUNCTION
        This call is similar to ObtainSemaphore(), except that it will not
        block if the semaphore could not be locked.

   INPUT
       signalSemaphore -- an initialized signal semaphore structure

   RESULT
        success -- TRUE if the semaphore was locked, false if some
            other task already possessed the semaphore.

   SEE ALSO
        ObtainSemaphore() ObtainSemaphoreShared(), ReleaseSemaphore(),
        exec/semaphores.h
```

exec.library Page 26

```
exec.library/AvailMem                                      exec.library/AvailMem

   NAME
        AvailMem -- memory available given certain requirements

   SYNOPSIS
        size = AvailMem(attributes)
        D0              D1

        ULONG AvailMem(ULONG);

   FUNCTION
        This function returns the amount of free memory given certain
        attributes.

        To find out what the largest block of a particular type is, add
        MEMF_LARGEST into the requirements argument.  Returning the largest
        block is a slow operation.

   WARNING
        Due to the effect of multitasking, the value returned may not
        actually be the amount of free memory available at that instant.

   INPUTS
        requirements - a requirements mask as specified in AllocMem.  Any
                       of the AllocMem bits are valid, as is MEMF_LARGEST
                       which returns the size of the largest block matching
                       the requirements.

   RESULT
        size - total free space remaining (or the largest free block).

   NOTE
        For V36 Exec, AvailMem(MEMF_LARGEST) does a consistency check on
        the memory list.  Alert AN_MemoryInsane will be pulled if any mismatch
        is noted.

   EXAMPLE
        AvailMem(MEMF_CHIP|MEMF_LARGEST);
        /* return size of largest available chip memory chunk */

   SEE ALSO
        exec/memory.h
```

```
exec.library/CacheClearE                          exec.library/CacheClearE

   NAME
        CacheClearE - Cache clearing with extended control (V37)

   SYNOPSIS
        CacheClearE(address,length,caches)
                    a0      d0     d1

        void CacheClearE(APTR,ULONG,ULONG);

   FUNCTION
        Flush out the contents of the CPU instruction and/or data caches.
        If dirty data cache lines are present, push them to memory first.

        Motorola CPUs have separate instruction and data caches.  A data
        write does not update the instruction cache.  If an instruction is
        written to memory or modified, the old instruction may still exist
        in the cache.  Before attempting to execute the code, a flush of
        the instruction cache is required.

        For most systems, the data cache is not updated by Direct Memory
        Access (DMA), or if some external factor changes shared memory.

        Caches must be cleared after *any* operation that could cause
        invalid or stale data.  The most common cases are DMA and modifying
        instructions using the processor.

        Some examples:
                        Self modifying code
                        Building Jump tables
                        Run-time code patches
                        Relocating code for use at different addresses.
                        Loading code from disk

   INPUTS
        address - Address to start the operation.  This may be rounded
                  due to hardware granularity.
        length  - Length of area to be cleared, or $FFFFFFFF to indicate all
                  addresses should be cleared.
        caches  - Bit flags to indicate what caches to affect.  The current
                  supported flags are:
                        CACRF_ClearI    ;Clear instruction cache
                        CACRF_ClearD    ;Clear data cache
                  All other bits are reserved for future definition.

   NOTES
        On systems with a copyback mode cache, any dirty data is pushed
        to memory as a part of this operation.

        Regardless of the length given, the function will determine the most
        efficient way to implement the operation.  For some cache systems,
        including the 68030, the overhead partially clearing a cache is often
        too great.  The entire cache may be cleared.

        For all current Amiga models, Chip memory is set with Instruction
        caching enabled, data caching disabled.  This prevents coherency
        conflicts with the blitter or other custom chip DMA.  Custom chip
        registers are marked as non-cacheable by the hardware.

        The system takes care of appropriately flushing the caches for normal
        operations.  The instruction cache is cleared by all calls that
        modify instructions, including LoadSeg(), MakeLibrary() and
        SetFunction().

   SEE ALSO
```

```
        exec/execbase.i, CacheControl, CacheClearU
```

```
exec.library/CacheClearU                                exec.library/CacheClearU

   NAME
        CacheClearU - User callable simple cache clearing (V37)

   SYNOPSIS
        CacheClearU()

        void CacheClearU();

   FUNCTION
        Flush out the contents of any CPU instruction and data caches.
        If dirty data cache lines are present, push them to memory first.

        Caches must be cleared after *any* operation that could cause
        invalid or stale data.  The most common cases are DMA and modifying
        instructions using the processor.  See the CacheClearE() autodoc
        for a more complete description.

        Some examples of when the cache needs clearing:
                        Self modifying code
                        Building Jump tables
                        Run-time code patches
                        Relocating code for use at different addresses.
                        Loading code from disk

   SEE ALSO
        exec/execbase.i, CacheControl, CacheClearE
```

```
exec.library/CacheControl                                exec.library/CacheControl

   NAME
        CacheControl - Instruction & data cache control

   SYNOPSIS
        oldBits = CacheControl(cacheBits,cacheMask)
        D0                     D0        D1

        ULONG CacheControl(ULONG,ULONG);

   FUNCTION
        This function provides global control of any instruction or data
        caches that may be connected to the system.  All settings are
        global -- per task control is not provided.

        The action taken by this function will depend on the type of
        CPU installed.  This function may be patched to support external
        caches, or different cache architectures.  In all cases the function
        will attempt to best emulate the provided settings.  Use of this
        function may save state specific to the caches involved.

        The list of supported settings is provided in the exec/execbase.i
        include file.  The bits currently defined map directly to the Motorola
        68030 CPU CACR register.  Alternate cache solutions may patch into
        the Exec cache functions.  Where possible, bits will be interpreted to
        have the same meaning on the installed cache.

   INPUTS
        cacheBits - new values for the bits specified in cacheMask.

        cacheMask - a mask with ones for all bits to be changed.

   RESULT
        oldBits   - the complete prior values for all settings.

   NOTE
        As a side effect, this function clears all caches.

   SEE ALSO
        exec/execbase.i, CacheClearU, CacheClearE
```

```
exec.library/Cause                                    exec.library/Cause

   NAME
       Cause -- cause a software interrupt

   SYNOPSIS
       Cause(interrupt)
             A1

       void Cause(struct Interrupt *);

   FUNCTION
        This function causes a software interrupt to occur.  If it is
        called from user mode (and processor level 0), the software
        interrupt will preempt the current task.  This call is often used
        by high-level hardware interrupts to defer medium-length processing
        down to a lower interrupt level.  Note that a software interrupt is
        still a real interrupt, and must obey the same restrictions on what
        system function it may call.

        Currently only 5 software interrupt priorities are implemented:
        -32, -16, 0, +16, and +32.  Priorities in between are truncated,
        values outside the -32/+32 range are not allowed.

   NOTE
        When setting up the Interrupt structure, set the node type to
        NT_INTERRUPT, or NT_UNKOWN.

   IMPLEMENTATION
        1> Checks if the node type is NT_SOFTINT.  If so does nothing since
           the softint is already pending.  No nest count is maintained.
        2> Sets the node type to NT_SOFTINT.
        3> Links into one of the 5 priority queues.
        4> Pokes the hardware interrupt bit used for softints.

        The node type returns to NT_INTERRUPT after removal from the list.

   INPUTS
        interrupt - pointer to a properly initialized interrupt node

   BUGS
        Unlike other Interrupts, SoftInts must preserve the value of A6.
```

```
exec.library/CheckIO                                  exec.library/CheckIO

   NAME
       CheckIO -- get the status of an IORequest

   SYNOPSIS
       result = CheckIO(iORequest)
       D0               A1

       BOOL CheckIO(struct IORequest *);

   FUNCTION
       This function determines the current state of an I/O request and
       returns FALSE if the I/O has not yet completed.  This function
       effectively hides the internals of the I/O completion mechanism.

       CheckIO() will NOT remove the returned IORequest from the reply port.
       This is best performed with WaitIO(). If the request has already
       completed, WaitIO() will return quickly. Use of the Remove()
       function is dangerous, since other tasks may still be adding things
       to your message port; a Disable() would be required.

       This function should NOT be used to busy loop (looping until IO is
       complete).  WaitIO() is provided for that purpose.

   INPUTS
       iORequest - pointer to an I/O request block

   RESULTS
       result - NULL if I/O is still in progress.  Otherwise
                D0 points to the IORequest block.

   NOTE
       CheckIO can hang if called on an IORequest that has never been used.
       This occurs if LN_TYPE of the IORequest is set to "NT_MESSAGE".
       Instead simply set LN_TYPE to 0.

   SEE ALSO
       DoIO, SendIO, WaitIO, AbortIO
```

exec.library

Page 33

```
exec.library/CloseDevice                               exec.library/CloseDevice

   NAME
        CloseDevice -- conclude access to a device

   SYNOPSIS
        CloseDevice(iORequest)
                    A1

        void CloseDevice(struct IORequest *);

   FUNCTION
        This function informs the device that access to a device/unit
        previously opened has been concluded.  The device may perform
        certain house-cleaning operations.

        The user must ensure that all outstanding IORequests have been
        returned before closing the device.  The AbortIO function can kill
        any stragglers.

        After a close, the I/O request structure is free to be reused.
        Starting with V36 exec it is safe to CloseDevice() with an
        IORequest that is either cleared to zeros, or failed to
        open.

   INPUTS
        iORequest - pointer to an I/O request structure

   SEE ALSO
        OpenDevice
```

exec.library

Page 34

```
exec.library/CloseLibrary                             exec.library/CloseLibrary

   NAME
        CloseLibrary -- conclude access to a library

   SYNOPSIS
        CloseLibrary(library)
                     A1

        void CloseLibrary(struct Library *);

   FUNCTION
        This function informs the system that access to the given library
        has been concluded.  The user must not reference the library or any
        function in the library after this close.

        Starting with V36, it is safe to pass a NULL instead of
        a library pointer.

   INPUTS
        library - pointer to a library node

   NOTE
        Library writers must pass a SegList pointer or NULL back from their
        open point.  This value is used by the system, and not visible as
        a return code from CloseLibrary.

   SEE ALSO
        OpenLibrary
```

```
exec.library/ColdReboot                              exec.library/ColdReboot

   NAME
       ColdReboot - reboot the Amiga (V36)

   SYNOPSIS
       ColdReboot()

       void ColdReboot(void);

   FUNCTION
       Reboot the machine.  All external memory and periperals will be
       RESET, and the machine will start its power up diagnostics.

       This function never returns.

   INPUT
       A chaotic pile of disoriented bits.

   RESULTS
       An altogether totally integrated living system.
```

```
exec.library/CopyMem                                    exec.library/CopyMem

   NAME
       CopyMem - general purpose memory copy function

   SYNOPSIS
       CopyMem( source, dest, size )
                A0      A1    D0

       void CopyMem(APTR,APTR,ULONG);

   FUNCTION
       CopyMem is a general purpose, fast memory copy function.  It can
       deal with arbitrary lengths, with its pointers on arbitrary
       alignments.  It attempts to optimize larger copies with more
       efficient copies, it uses byte copies for small moves, parts of
       larger copies, or the entire copy if the source and destination are
       misaligned with respect to each other.

       Arbitrary overlapping copies are not supported.

       The internal implementation of this function will change from
       system to system, and may be implemented via hardware DMA.

   INPUTS
       source - a pointer to the source data region
       dest  - a pointer to the destination data region
       size  - the size (in bytes) of the memory area.  Zero copies
               zero bytes

   SEE ALSO
       CopyMemQuick
```

```
exec.library/CopyMemQuick                               exec.library/CopyMemQuick

   NAME
        CopyMemQuick - optimized memory copy function

   SYNOPSIS
        CopyMemQuick( source, dest, size )
                      A0      A1    D0

        void CopyMem(ULONG *,ULONG *,ULONG);

   FUNCTION
        CopyMemQuick is a highly optimized memory copy function, with
        restrictions on the size and alignment of its arguments. Both the
        source and destination pointers must be longword aligned.  In
        addition, the size must be an integral number of longwords (e.g.
        the size must be evenly divisible by four).

        Arbitrary overlapping copies are not supported.

        The internal implementation of this function will change from system
        to system, and may be implemented via hardware DMA.

   INPUTS
        source - a pointer to the source data region, long aligned
        dest -  a pointer to the destination data region, long aligned
        size -  the size (in bytes) of the memory area.  Zero copies
                zero bytes.

   SEE ALSO
        CopyMem
```

```
exec.library/CreateIORequest                         exec.library/CreateIORequest

   NAME
        CreateIORequest() -- create an IORequest structure  (V36)

   SYNOPSIS
        ioReq = CreateIORequest( ioReplyPort, size );
                                 A0           D0

        struct IORequest *CreateIORequest(struct MsgPort *, ULONG);

   FUNCTION
        Allocates memory for and initializes a new IO request block
        of a user-specified number of bytes.  The number of bytes
        must be at least as large as a "struct Message".

   INPUTS
        ioReplyPort - Pointer to a port for replies (an initialized message
                port, as created by CreateMsgPort() ).  If NULL, this
                function fails.
        size - the size of the IO request to be created.

   RESULT
        ioReq - A pointer to the new IORequest block, or NULL.

   SEE ALSO
        DeleteIORequest, CreateMsgPort(), amiga.lib/CreateExtIO()
```

```
exec.library/CreateMsgPort                              exec.library/CreateMsgPort

   NAME
        CreateMsgPort - Allocate and initialize a new message port  (V36)

   SYNOPSIS
        CreateMsgPort()

        struct MsgPort * CreateMsgPort(void);

   FUNCTION
        Allocates and initializes a new message port.  The message list
        of the new port will be prepared for use (via NewList).  A signal
        bit will be allocated, and the port will be set to signal your
        task when a message arrives (PA_SIGNAL).

        You *must* use DeleteMsgPort() to delete ports created with
        CreateMsgPort()!

   RESULT
        MsgPort - A new MsgPort structure ready for use, or NULL if out of
                memory or signals.  If you wish to add this port to the public
                port list, fill in the ln_Name and ln_Pri fields, then call
                AddPort().  Don't forget RemPort()!

   SEE ALSO
        DeleteMsgPort(), exec/AddPort(), exec/ports.h, amiga.lib/CreatePort()
```

```
exec.library/Deallocate                                    exec.library/Deallocate

   NAME
        Deallocate -- deallocate a block of memory

   SYNOPSIS
        Deallocate(memHeader, memoryBlock, byteSize)
                   A0         A1           D0

        void Deallocate(struct MemHeader *,APTR,ULONG);

   FUNCTION
        This function deallocates memory by returning it to the appropriate
        private free memory pool.  This function can be used to free an
        entire block allocated with the above function, or it can be used
        to free a sub-block of a previously allocated block.  Sub-blocks
        must be an even multiple of the memory chunk size (currently 8
        bytes).

        This function can even be used to add a new free region to an
        existing MemHeader, however the extent pointers in the MemHeader
        will no longer be valid.

        If memoryBlock is not on a block boundary (MEM_BLOCKSIZE) then it
        will be rounded down in a manner compatible with Allocate().  Note
        that this will work correctly with all the memory allocation
        functions, but may cause surprises if one is freeing only part of a
        region.  The size of the block will be rounded up, so the freed
        block will fill to an even memory block boundary.

   INPUTS
        memHeader - points to the memory header this block is part of.
        memoryBlock - address of memory block to free.
        byteSize - the size of the block in bytes. If NULL, nothing
                   happens.

   SEE ALSO
        Allocate, exec/memory.h
```

exec.library

Page 41

```
exec.library/Debug                                         exec.library/Debug

   NAME
        Debug -- run the system debugger

   SYNOPSIS
        Debug(flags)
              D0

        void Debug(ULONG);

   FUNCTION
        This function calls the system debugger.  By default this debugger
        is "ROM-WACK".  Other debuggers are encouraged to take over this
        entry point (via SetFunction()) so that when an application calls
        Debug(), the alternative debugger will get control.  Currently a
        zero is passed to allow future expansion.

   NOTE
        The Debug() call may be made when the system is in a questionable
        state; if you have a SetFunction() patch, make few assumptions, be
        prepared for Supervisor mode, and be aware of differences in the
        Motorola stack frames on the 68000,'10,'20, and '30.

   SEE ALSO
        SetFunction
        your favorite debugger's manual
        the ROM-WACK chapter of the ROM Kernel Manual
```

exec.library

Page 42

```
exec.library/DeleteIORequest                     exec.library/DeleteIORequest

   NAME
        DeleteIORequest() - Free a request made by CreateIORequest()  (V36)

   SYNOPSIS
        DeleteIORequest( ioReq );
                         a0

        void DeleteIORequest(struct IORequest *);

   FUNCTION
        Frees up an IO request as allocated by CreateIORequest().

   INPUTS
        ioReq - A pointer to the IORequest block to be freed, or NULL.
                This function uses the mn_Length field to determine how
                much memory to free.

   SEE ALSO
        CreateIORequest(), amiga.lib/DeleteExtIO()
```

```
exec.library/DeleteMsgPort                            exec.library/DeleteMsgPort

   NAME
        DeleteMsgPort - Free a message port created by CreateMsgPort  (V36)

   SYNOPSIS
        DeleteMsgPort(msgPort)
                      a0

        void DeleteMsgPort(struct MsgPort *);

   FUNCTION
        Frees a message port created by CreateMsgPort().  All messages that
        may have been attached to this port must have already been
        replied to.

   INPUTS
        msgPort - A message port.  NULL for no action.

   SEE ALSO
        CreateMsgPort(), amiga.lib/DeletePort()
```

```
exec.library/Disable                                        exec.library/Disable

   NAME
        Disable -- disable interrupt processing.

   SYNOPSIS
        Disable();

        void Disable(void);

   FUNCTION
        Prevents interrupts from being handled by the system, until a
        matching Enable() is executed.  Disable() implies Forbid().

        DO NOT USE THIS CALL WITHOUT GOOD JUSTIFICATION.  THIS CALL IS
        VERY DANGEROUS!

   RESULTS
        All interrupt processing is deferred until the task executing makes
        a call to Enable() or is placed in a wait state.  Normal task
        rescheduling does not occur while interrupts are disabled.  In order
        to restore normal interrupt processing, the programmer must execute
        exactly one call to Enable() for every call to Disable().

        IMPORTANT REMINDER:

        It is important to remember that there is a danger in using
        disabled sections.  Disabling interrupts for more than ~250
        microseconds will prevent vital system functions (especially serial
        I/0) from operating in a normal fashion.

        Think twice before using Disable(), then think once more.
        After all that, think again.  With enough thought, the need
        for a Disable() can often be eliminated.  For the user of many
        device drivers, a write to disable *only* the particular interrupt
        of interest can replace a Disable().  For example:
                        MOVE.W  #INTF_PORTS,_intena
        Do not use a macro for Disable(), insist on the real thing.

        This call may be made from interrupts, it will have the effect
        of locking out all higher-level interrupts (lower-level interrupts
        are automatically disabled by the CPU).

        Note: In the event of a task entering a Wait() after disabling
              interrupts, the system "breaks" the disabled state and runs
              normally until the task which called Disable() is rescheduled.

   NOTE
        This call is guaranteed to preserve all registers.

  SEE ALSO
        Forbid, Permit, Enable
```

exec.library Page 45

```
exec.library/DoIO                                      exec.library/DoIO

   NAME
       DoIO -- perform an I/O command and wait for completion

   SYNOPSIS
       error = DoIO(iORequest)
       D0           A1

       BYTE DoIO(struct IORequest *);

   FUNCTION
       This function requests a device driver to perform the I/O command
       specified in the I/O request.  This function will always wait until
       the I/O request is fully complete.

       DoIO() handles all the details, including Quick I/O, waiting for
       the request, clearing signal bits, and removing the reply message.

   IMPLEMENTATION
       This function first tries to complete the IO via the "Quick I/O"
       mechanism.  The io_Flags field is always set to IOF_QUICK (0x01)
       before the internal device call.

       The LN_TYPE field is used internally to flag completion.  Active
       requests have type NT_MESSAGE.  Requests that have been replied
       have type NT_REPLYMSG.  It is illegal to start IO using a
       still active IORequest, or a request with type NT_REPLYMSG.

   INPUTS
       iORequest - pointer to an IORequest initialized by OpenDevice()

   RESULTS
       error - a sign-extended copy of the io_Error field of the
               IORequest.  Most device commands require that the error
               return be checked.

   SEE ALSO
       SendIO, CheckIO, WaitIO, AbortIO, amiga.lib/BeginIO
```

exec.library Page 46

```
exec.library/Enable                                          exec.library/Enable

   NAME
       Enable -- permit system interrupts to resume.

   SYNOPSIS
       Enable();

       void Enable(void);

   FUNCTION
       Allow system interrupts to again occur normally, after a matching
       Disable() has been executed.

   RESULTS
       Interrupt processing is restored to normal operation. The
       programmer must execute exactly one call to Enable() for every call
       to Disable().

    NOTE
       This call is guaranteed to preserve all registers.

   SEE ALSO
       Forbid, Permit, Disable
```

```
exec.library/Enqueue                                    exec.library/Enqueue

   NAME
       Enqueue -- insert or append node to a system queue

   SYNOPSIS
       Enqueue(list, node)
               A0    A1

       void Enqueue(struct List *, struct Node *);

   FUNCTION
       Insert or append a node into a system queue.  The insert is
       performed based on the node priority -- it will keep the list
       properly sorted.  New nodes will be inserted in front of the first
       node with a lower priority.   Hence a FIFO queue for nodes of equal
       priority

   WARNING
       This function does not arbitrate for access to the list.  The
       calling task must be the owner of the involved list.

   INPUTS
       list - a pointer to the system queue header
       node - the node to enqueue.  This must be a full featured list
              with type, priority and name fields.

   SEE ALSO
       AddHead, AddTail, Insert, Remove, RemHead, RemTail
```

```
exec.library/FindName                                  exec.library/FindName

   NAME
       FindName -- find a system list node with a given name

   SYNOPSIS
       node = FindName(start, name)
       D0,Z            A0     A1

       struct Node *FindName(struct List *, STRPTR);

   FUNCTION
       Traverse a system list until a node with the given name is found.
       To find multiple occurrences of a string, this function may be
       called with a node starting point.

       No arbitration is done for access to the list!  If multiple tasks
       access the same list, an arbitration mechanism such as
       SignalSemaphores must be used.

   INPUTS
       start - a list header or a list node to start the search
            (if node, this one is skipped)
       name - a pointer to a name string terminated with NULL

   RESULTS
       node - a pointer to the node with the same name else
           zero to indicate that the string was not found.
```

exec.library Page 49

```
exec.library/FindPort                                exec.library/FindPort

   NAME
        FindPort -- find a given system message port

   SYNOPSIS
        port = FindPort(name)
        D0              A1

        struct MsgPort *FindPort(STRPTR);

   FUNCTION
        This function will search the system message port list for a port
        with the given name.  The first port matching this name will be
        returned.  No arbitration of the port list is done.  This function
        MUST be protected with A Forbid()/Permit() pair!

   EXAMPLE
        #include <exec/types.h>
        struct MsgPort *FindPort();

        ULONG SafePutToPort(message, portname)
        struct Message *message;
        STRPTR          portname;
        {
        struct MsgPort *port;

            Forbid();
                port = FindPort(portname);
                if (port)
                    PutMsg(port,message);
            Permit();
            return((ULONG)port); /* If zero, the port has gone away */
        }

   INPUT
        name - name of the port to find

   RETURN
        port - a pointer to the message port, or zero if
                not found.
```

exec.library Page 50

```
exec.library/FindResident                        exec.library/FindResident

   NAME
        FindResident - find a resident module by name

   SYNOPSIS
        resident = FindResident(name)
        D0                      A1

        struct Resident *FindResident(STRPTR);

   FUNCTION
        Find the resident tag with the given name.  If found return a
        pointer to the resident tag structure, else return zero.

        Resident modules are used by the system to pull all its parts
        together at startup.  Resident tags are also found in disk based
        devices and libraries.

   INPUTS
        name - pointer to name string

    RESULT
        resident - pointer to the resident tag structure or
                zero if none found.

    SEE ALSO
        exec/resident.h
```

```
exec.library/FindSemaphore                         exec.library/FindSemaphore

   NAME
        FindSemaphore -- find a given system signal semaphore

   SYNOPSIS
        signalSemaphore = FindSemaphore(name)
        D0                              A1

        struct SignalSemaphore *FindSemaphore(STRPTR);

   FUNCTION
        This function will search the system signal semaphore list for a
        semaphore with the given name.  The first semaphore matching this
        name will be returned.

        This function does not arbitrate for access to the semaphore list,
        surround the call with a Forbid()/Permit() pair.

   INPUT
        name - name of the semaphore to find

   RESULT
        semaphore - a pointer to the signal semaphore, or zero if not
                    found.
```

```
exec.library/FindTask                                   exec.library/FindTask

   NAME
        FindTask -- find a task with the given name or find oneself

   SYNOPSIS
        task = FindTask(name)
        D0              A1

        struct Task *FindTask(STRPTR);

   FUNCTION
        This function will check all task queues for a task with the given
        name, and return a pointer to its task control block.  If a NULL
        name pointer is given a pointer to the current task will be
        returned.

        Finding oneself with a NULL for the name is very quick.  Finding a
        task by name is very system expensive, and will disable interrupts
        for a long time.  Since a task may remove itself at any time,
        a Forbid()/Permit() pair may be needed to ensure the pointer
        returned by FindTask() is still valid when used.

   INPUT
        name - pointer to a name string

   RESULT
        task - pointer to the task (or Process)
```

exec.library Page 53

```
exec.library/Forbid                                     exec.library/Forbid

   NAME
       Forbid -- forbid task rescheduling.

   SYNOPSIS
       Forbid()

       void Forbid(void);

   FUNCTION
       Prevents other tasks from being scheduled to run by the dispatcher,
       until a matching Permit() is executed, or this task is scheduled to
       Wait().  Interrupts are NOT disabled.

       DO NOT USE THIS CALL WITHOUT GOOD JUSTIFICATION.  THIS CALL IS
       DANGEROUS!

   RESULTS
       The current task will not be rescheduled as long as it is ready to
       run.  In the event that the current task enters a wait state, other
       tasks may be scheduled.  Upon return from the wait state, the original
       task will continue to run without disturbing the Forbid().

       Calls to Forbid() nest. In order to restore normal task rescheduling,
       the programmer must execute exactly one call to Permit() for every
       call to Forbid().

   WARNING
       In the event of a task entering a Wait() after a Forbid(), the system
       "breaks" the forbidden state and runs normally until the task which
       called Forbid() is rescheduled.  If caution is not taken, this can
       cause subtle bugs, since any device or DOS call will (in effect)
       cause your task to wait.

       Forbid() is not useful or safe from within interrupt code
       (All interrupts are always higher priority than tasks, and
       interrupts are allowed to break a Forbid()).

   NOTE
       This call is guaranteed to preserve all registers.

   SEE ALSO
       Permit, Disable, ObtainSemaphore, ObtainSemaphoreShared
```

exec.library Page 54

```
exec.library/FreeEntry                                           exec.library/FreeEntry

   NAME
       FreeEntry -- free many regions of memory

   SYNOPSIS
       FreeEntry(memList)
                 A0
       void FreeEntry(struct MemList *);

   FUNCTION
       This function takes a memList structure (as returned by AllocEntry)
       and frees all the entries.

   INPUTS
       memList -- pointer to structure filled in with MemEntry
                  structures

   SEE ALSO
       AllocEntry
```

```
exec.library/FreeMem                                    exec.library/FreeMem

   NAME
        FreeMem -- deallocate with knowledge

   SYNOPSIS
        FreeMem(memoryBlock, byteSize)
                A1           D0

        void FreeMem(void *,ULONG);

   FUNCTION
        Free a region of memory, returning it to the system pool from which
        it came.  Freeing partial blocks back into the system pool is
        unwise.

   NOTE
        If a block of memory is freed twice, the system will Guru. The
        Alert is AN_FreeTwice ($01000009).   If you pass the wrong pointer,
        you will probably see AN_MemCorrupt $01000005.  Future versions may
        add more sanity checks to the memory lists.

   INPUTS
        memoryBlock - pointer to the memory block to free
        byteSize - the size of the desired block in bytes.  (The operating
                system will automatically round this number to a multiple of
                the system memory chunk size)

   SEE ALSO
        AllocMem
```

```
exec.library/FreeSignal                              exec.library/FreeSignal

   NAME
        FreeSignal -- free a signal bit

   SYNOPSIS
        FreeSignal(signalNum)
                   D0

        FreeSignal(BYTE);

   FUNCTION
        This function frees a previously allocated signal bit for reuse.
        This call must be performed while running in the same task in which
        the signal was allocated.

   WARNING
        Signals may not be allocated or freed from exception handling code.

   NOTE
        Starting with V37, an attempt to free signal -1 is harmless.

   INPUTS
        signalNum - the signal number to free {0..31}.
```

exec.library Page 57

```
exec.library/FreeTrap                                    exec.library/FreeTrap

   NAME
       FreeTrap -- free a processor trap

   SYNOPSIS
       FreeTrap(trapNum)
                D0

       void FreeTrap(ULONG);

   FUNCTION
       This function frees a previously allocated trap number for reuse.
       This call must be performed while running in the same task in which
       the trap was allocated.

   WARNING
       Traps may not be allocated or freed from exception handling code.

   INPUTS
       trapNum - the trap number to free {of 0..15}
```

exec.library Page 58

```
exec.library/FreeVec                                              exec.library/FreeVec

   NAME
       FreeVec -- return AllocVec() memory to the system  (V36)

   SYNOPSIS
       FreeVec(memoryBlock)
               A1

       void FreeVec(void *);

   FUNCTION
       Free an allocation made by the AllocVec() call.  The memory will
       be returned to the system pool from which it came.

   NOTE
       If a block of memory is freed twice, the system will Guru. The
       Alert is AN_FreeTwice ($01000009).   If you pass the wrong pointer,
       you will probably see AN_MemCorrupt $01000005.  Future versions may
       add more sanity checks to the memory lists.

   INPUTS
       memoryBlock - pointer to the memory block to free, or NULL.

   SEE ALSO
       AllocVec
```

```
exec.library/GetCC                                        exec.library/GetCC

   NAME
       GetCC -- get condition codes in a 68010 compatible way.

   SYNOPSIS
       conditions = GetCC()
         D0

       UWORD = GetCC(void);

   FUNCTION
       The 68000 processor has a "MOVE SR,<ea>" instruction which gets a
       copy of the processor condition codes.

       On the 68010,20 and 30 CPUs, "MOVE SR,<ea>" is privileged.  User
       code will trap if it is attempted.  These processors need to use
       the "MOVE CCR,<ea>" instruction instead.

       This function provides a means of obtaining the CPU condition codes
       in a manner that will make upgrades transparent.  This function is
       VERY short and quick.

   RESULTS
       conditions - the 680XX condition codes

    NOTE
       This call is guaranteed to preserve all registers.  This function
       may be implemented as code right in the jump table.
```

```
exec.library/GetMsg                                      exec.library/GetMsg

   NAME
       GetMsg -- get next message from a message port

   SYNOPSIS
       message = GetMsg(port)
       D0               A0

       struct Message *GetMsg(struct MsgPort *);

   FUNCTION
       This function receives a message from a given message port. It
       provides a fast, non-copying message receiving mechanism. The
       received message is removed from the message port.

       This function will not wait.  If a message is not present this
       function will return zero.  If a program must wait for a message,
       it can Wait() on the signal specified for the port or use the
       WaitPort() function.  There can only be one task waiting for any
       given port.

       Getting a message does not imply to the sender that the message is
       free to be reused by the sender.  When the receiver is finished
       with the message, it may ReplyMsg() it back to the sender.


       Getting a signal does NOT always imply a message is ready.  More
       than one message may arrive per signal, and signals may show up
       without messages.  Typically you must loop to GetMsg() until it
       returns zero, then Wait() or WaitPort().

   INPUT
       port - a pointer to the receiver message port

   RESULT
       message - a pointer to the first message available.  If
                 there are no messages, return zero.
                 Callers must be prepared for zero at any time.

   SEE ALSO
       PutMsg, ReplyMsg, WaitPort, Wait, exec/ports.h
```

```
exec.library/InitCode                                   exec.library/InitCode

   NAME
        InitCode - initialize resident code modules (internal function)

   SYNOPSIS
        InitCode(startClass, version)
                 D0          D1

   FUNCTION
        (This function may be ignored by application programmers)

        Call InitResident() for all resident modules in the ResModules array
        with the given startClass and with versions equal or greater than
        that specified.  The segList parameter is passed as zero.

        Resident modules are used by the system to pull all its parts
        together at startup.  Modules are initialized in a prioritized order.

        Modules that do not have a startclass should be of priority -120.
        RTF_AFTERDOS modues should start at -100 (working down).

   INPUTS
        startClass - the class of code to be initialized:
                BITDEF RT,COLDSTART,0
                BITDEF RT,SINGLETASK,1  ;ExecBase->ThisTask==0 (V36 only)
                BITDEF RT,AFTERDOS,2    ;(V36 only)
        version - a major version number

    SEE ALSO
        ResidentTag (RT) structure definition (resident.h)
```

```
exec.library/InitResident                           exec.library/InitResident

   NAME
        InitResident - initialize resident module

   SYNOPSIS
        InitResident(resident, segList)
                     A1        D1
        void InitResident(struct Resident *,ULONG);

   FUNCTION
        Initialize a ROMTag.  ROMTags are used to link system modules
        together.  Each disk based device or library must contain a
        ROMTag structure in the first code hunk.

        Once the validity of the ROMTag is verified, the RT_INIT pointer
        is jumped to  with the following registers:
                D0 = 0
                A0 = segList
                A6 = ExecBase

   AUTOINIT FEATURE
        An automatic method of library/device base and vector table
        initialization is also provided by InitResident().  The initial code
        hunk of the library or device should contain "MOVEQ #-1,d0; RTS;".
        Following that must be an initialized Resident structure with
        RTF_AUTOINIT set in rt_Flags, and an rt_Init pointer which points
        to four longwords.  These four longwords will be used in a call
        to MakeLibrary();

            - The size of your library/device base structure including initial
              Library or Device structure.

            - A pointer to a longword table of standard, then library
              specific function offsets, terminated with -1L.
              (short format offsets are also acceptable)

            - Pointer to data table in exec/InitStruct format for
              initialization of Library or Device structure.

            - Pointer to library initialization function, or NULL.
                Calling sequence:
                        D0 = library base
                        A0 = segList
                        A6 = ExecBase
              This function must return in D0 the library/device base to be
              linked into the library/device list.  If the initialization
              function fails, the device memory must be manually deallocated,
              then NULL returned in D0.

   SEE ALSO
        exec/resident.i
```

```
exec.library/InitSemaphore                           exec.library/InitSemaphore

   NAME
       InitSemaphore -- initialize a signal semaphore

   SYNOPSIS
       InitSemaphore(signalSemaphore)
                     A0

       void InitSemaphore(struct SignalSemaphore *);

   FUNCTION
       This function initializes a signal semaphore and prepares it for
       use.  It does not allocate anything, but does initialize list
       pointers and the semaphore counters.

       Semaphores are often used to protect critical data structures
       or hardware that can only be accessed by one task at a time.
       After initialization, the address of the SignalSemaphore may be
       made available to any number of tasks.  Typically a task will
       try to ObtainSemaphore(), passing this address in.  If no other
       task owns the semaphore, then the call will lock and return
       quickly.  If more tasks try to ObtainSemaphore(), they will
       be put to sleep.  When the owner of the semaphore releases
       it, the next waiter in turn will be woken up.

       Semaphores are often preferable to the old-style Forbid()/Permit()
       type arbitration.  With Forbid()/Permit() *all* other tasks are
       prevented from running.  With semaphores, only those tasks that
       need access to whatever the semaphore protects are subject
       to waiting.

   INPUT
       signalSemaphore -- a signal semaphore structure (with all fields
                          set to zero before the call)

   SEE ALSO
       ObtainSemaphore(), ObtainSemaphoreShared(), AttemptSemaphore(),
       ReleaseSemaphore(), exec/semaphores.h
```

```
exec.library/InitStruct                                 exec.library/InitStruct

   NAME
       InitStruct - initialize memory from a table

   SYNOPSIS
       InitStruct(initTable, memory, size);
                  A1         A2      D0

       void InitStruct(struct InitStruct *, APTR, ULONG);

   FUNCTION
       Clear a memory area, then set up default values according to
       the data and offset values in the initTable.  Typically only assembly
       programs take advantage of this function, and only with the macros
       defined in "exec/initializers.i".

       The initialization table has byte commands to

            |a    ||byte|      |given||byte|         |once         |
       load |count||word| into |next ||rptr| offset, |repetitively |
                   |long|

       Not all combinations are supported.  The offset, when specified, is
       relative to the memory pointer provided (Memory), and is initially
       zero.  The initialization data (InitTable) contains byte commands
       whose 8 bits are interpreted as follows:

       ddssnnnn
           dd  the destination type (and size):
               00  no offset, use next destination, nnnn is count
               01  no offset, use next destination, nnnn is repeat
               10  destination offset is in the next byte, nnnn is count
               11  destination offset is in the next 24-bits, nnnn is count
           ss  the size and location of the source:
               00  long, from the next two aligned words
               01  word, from the next aligned word
               10  byte, from the next byte
               11  ERROR - will cause an ALERT (see below)
         nnnn  the count or repeat:
            count  the (number+1) of source items to copy
           repeat  the source is copied (number+1) times.

       initTable commands are always read from the next even byte. Given
       destination offsets are always relative to the memory pointer (A2).

       The command %00000000 ends the InitTable stream: use %00010001 if you
       really want to copy one longword without a new offset.

       24 bit APTR not supported for 68020 compatibility -- use long.

   INPUTS
       initTable - the beginning of the commands and data to init
               Memory with.  Must be on an even boundary unless only
               byte initialization is done.  End table with "dc.b 0"
               or "dc.w 0".
       memory - the beginning of the memory to initialize.  Must be
               on an even boundary if size is specified.
       size - the size of memory, which is used to clear it before
               initializing it via the initTable.  If Size is zero,
               memory is not cleared before initializing.

               size must be an even number.

    SEE ALSO
       exec/initializers.i
```

exec.library Page 65

```
exec.library/Insert                                        exec.library/Insert

   NAME
        Insert -- insert a node into a list

   SYNOPSIS
        Insert(list, node, listNode)
               A0    A1    A2

        void Insert(struct List *, struct Node *, struct Node *);

   FUNCTION
        Insert a node into a doubly linked list AFTER a given node
        position.  Insertion at the head of a list is possible by passing a
        zero value for listNode, though the AddHead function is slightly
        faster for that special case.

   WARNING
        This function does not arbitrate for access to the list.  The
        calling task must be the owner of the involved list.

   INPUTS
        list - a pointer to the target list header
        node - the node to insert
        listNode - the node after which to insert

    SEE ALSO
        AddHead, AddTail, Enqueue, RemHead, Remove, RemTail
```

exec.library Page 66

```
exec.library/MakeFunctions                          exec.library/MakeFunctions

   NAME
        MakeFunctions -- construct a function jump table

   SYNOPSIS
        tableSize = MakeFunctions(target, functionArray, funcDispBase)
        D0                        A0      A1             A2

        ULONG MakeFunctions(APTR,APTR,APTR);

   FUNCTION
        A low level function used by MakeLibrary to build jump tables of
        the type used by libraries, devices and resources.  It allows the
        table to be built anywhere in memory, and can be used both for
        initialization and replacement. This function also supports function
        pointer compression by expanding relative displacements into absolute
        pointers.

        The processor instruction cache is cleared after the table building.

   INPUT
        destination - the target address for the high memory end of the
                function jump table.  Typically this will be the library
                base pointer.

        functionArray - pointer to an array of function pointers or
                function displacements.  If funcDispBase is zero, the array
                is assumed to contain absolute pointers to functions. If
                funcDispBase is not zero, then the array is assumed to
                contain word displacements to functions.  In both cases,
                the array is terminated by a -1 (of the same size as the
                actual entry.

        funcDispBase - pointer to the base about which all function
                displacements are relative.  If zero, then the function
                array contains absolute pointers.

   RESULT
        tableSize - size of the new table in bytes (for LIB_NEGSIZE).

   SEE ALSO
        exec/MakeLibrary
```

```
exec.library/MakeLibrary                              exec.library/MakeLibrary

   NAME
        MakeLibrary -- construct a library

   SYNOPSIS
        library = MakeLibrary(vectors, structure, init, dSize, segList)
        D0                    A0       A1         A2    D0     D1

        struct Library *MakeLibrary
                              (APTR,struct InitStruct *,APTR,ULONG,BPTR);

   FUNCTION
        This function is used for constructing a library vector and data
        area.  The same call is used to make devices.  Space for the library
        is allocated from the system's free memory pool.  The data portion of
        the library is initialized.  init may point to a library specific
        entry point.

   NOTE
        Starting with V36, the library base is longword adjusted.  The
        lib_PosSize and lib_NegSize fields of the library structure are
        adjusted to match.

   INPUTS
        vectors - pointer to an array of function pointers or function
                displacements.  If the first word of the array is -1, then
                the array contains relative word displacements (based off
                of vectors); otherwise, the array contains absolute
                function pointers. The vector list is terminated by a -1
                (of the same size as the pointers).

        structure - points to an "InitStruct" data region.  If NULL,
                then it will not be used.

        init -  If non-NULL, an entry point that will be called before adding
                the library to the system.  Registers are as follows:
                        d0 = libAddr    ;Your Library Address
                        a0 = segList    ;Your AmigaDOS segment list
                        a6 = ExecBase   ;Address of exec.library
                The result of the init function must be the library address,
                or NULL for failure.   If NULL, the init point must manually
                deallocate the library base memory (based on the sizes stored
                in lib_PosSize and lib_NegSize).

        dSize - the size of the library data area, including the
                standard library node data.  This must be at leas
                sizeof(struct Library).

       segList - pointer to an AmigaDOS SegList (segment list).
                 This is passed to a library's init code, and is used later
                 for removing the library from memory.

   RESULT
        library - the reference address of the library.  This is the
                  address used in references to the library, not the
                  beginning of the memory area allocated.  If the library
                  vector table require more system memory than is
                  available, this function will return NULL.

   SEE ALSO
        InitStruct, InitResident, exec/initializers.i
```

```
exec.library/ObtainSemaphore                      exec.library/ObtainSemaphore

   NAME
        ObtainSemaphore -- gain exclusive access to a semaphore

   SYNOPSIS
        ObtainSemaphore(signalSemaphore)
                        A0

        void ObtainSemaphore(struct SignalSemaphore *);

   FUNCTION
        Signal semaphores are used to gain exclusive access to an object.
        ObtainSemaphore is the call used to gain this access.  If another
        user currently has the semaphore locked the call will block until
        the object is available.

        If the current task already has locked the semaphore and attempts to
        lock it again the call will still succeed.  A "nesting count" is
        incremented each time the current owning task of the semaphore calls
        ObtainSemaphore().  This counter is decremented each time
        ReleaseSemaphore() is called.  When the counter returns to zero the
        semaphore is actually released, and the next waiting task is called.

        A queue of waiting tasks is maintained on the stacks of the waiting
        tasks.  Each will be called in turn as soon as the current task
        releases the semaphore.

        Signal Semaphores are different than Procure()/Vacate() semaphores.
        The former requires less CPU time, especially if the semaphore is
        not currently locked.  They require very little set up and user
        thought.  The latter flavor of semaphore make no assumptions about
        how they are used -- they are completely general.  Unfortunately
        they are not as efficient as signal semaphores, and require the
        locker to have done some setup before doing the call.

   INPUT
       signalSemaphore -- an initialized signal semaphore structure

   NOTE
        This function preserves all registers.

   SEE ALSO
        ObtainSemaphoreShared(), InitSemaphore(), ReleaseSemaphore(),
        AttemptSemaphore(), ObtainSemaphoreList()
```

```
exec.library/ObtainSemaphoreList              exec.library/ObtainSemaphoreList

   NAME
        ObtainSemaphoreList -- get a list of semaphores.

   SYNOPSIS
        ObtainSemaphoreList(list)
                            A0

        void ObtainSemaphoreList(struct List *);

   FUNCTION
        Signal semaphores may be linked together into a list. This function
        takes a list of these semaphores and attempts to lock all of them at
        once. This call is preferable to applying ObtainSemaphore() to each
        element in the list because it attempts to lock all the elements
        simultaneously, and won't deadlock if someone is attempting to lock
        in some other order.

        This function assumes that only one task at a time will attempt to
        lock the entire list of semaphores.  In other words, there needs to
        be a higher level lock (perhaps another signal semaphore...) that is
        used before someone attempts to lock the semaphore list via
        ObtainSemaphoreList().

        Note that deadlocks may result if this call is used AND someone
        attempts to use ObtainSemaphore() to lock more than one semaphore on
        the list.  If you wish to lock more than semaphore (but not all of
        them) then you should obtain the higher level lock (see above)

   INPUT
       list -- a list of signal semaphores

   SEE ALSO
        InitSemaphore(), ReleaseSemaphoreList()
```

```
exec.library/ObtainSemaphoreShared          exec.library/ObtainSemaphoreShared

    NAME
        ObtainSemaphoreShared -- gain shared access to a semaphore (V36)

    SYNOPSIS
        ObtainSemaphoreShared(signalSemaphore)
                              a0

    FUNCTION
        A lock on a signal semaphore may either be exclusive, or shared.
        Exclusive locks are granted by the ObtainSemaphore() and
        AttemptSemaphore() functions.  Shared locks are granted by
        ObtainSemaphoreShared().  Calls may be nested.

        Any number of tasks may simultaneously hold a shared lock on a
        semaphore.  Only one task may hold an exclusive lock.  A typical
        application is a list that is often read, but only occasionally
        written to.

        Any exlusive locker will be held off until all shared lockers
        release the semaphore.  Likewise, if an exlusive lock is held,
        all potential shared lockers will block until the exclusive lock
        is released.  All shared lockers are restarted at the same time.

    EXAMPLE
                ObtainSemaphoreShared(ss);
                /* read data */
                ReleaseSemaohore(ss);

                ObtainSemaphore(ss);
                /* modify data */
                ReleaseSemaohore(ss);

    NOTES
        While this funciton was added for V36, the feature magically works
        with all older semaphore structures.

        A task owning a shared lock must not attempt to get an exclusive
        lock on the same semaphore.

    INPUT
        signalSemaphore -- an initialized signal semaphore structure

   NOTE
        This function preserves all registers.

    RESULT

    SEE ALSO
        InitSemaphore(), ReleaseSemaphore()
```

exec.library

```
exec.library/OldOpenLibrary                        exec.library/OldOpenLibrary

   NAME
       OldOpenLibrary -- obsolete OpenLibrary

   SYNOPSIS
       library = OldOpenLibrary(libName)
       D0                       A1

       struct Library *OldOpenLibrary(APTR);

   FUNCTION
       The 1.0 release of the Amiga system had an incorrect version of
       OpenLibrary that did not check the version number during the
       library open.  This obsolete function is provided so that object
       code compiled using a 1.0 system will still run.

       This exactly the same as "OpenLibrary(libName,0L);"

   INPUTS
       libName - the name of the library to open

   RESULTS
       library - a library pointer for a successful open, else zero

   SEE ALSO
       CloseLibrary
```

exec.library

```
exec.library/OpenDevice                                exec.library/OpenDevice

   NAME
       OpenDevice -- gain access to a device

   SYNOPSIS
       error = OpenDevice(devName, unitNumber, iORequest, flags)
       D0                 A0       D0          A1         D1

       BYTE OpenDevice(STRPTR,ULONG,struct IORequest *,ULONG);

   FUNCTION
       This function opens the named device/unit and initializes the given
       I/O request block.  Specific documentation on opening procedures
       may come with certain devices.

       The device may exist in memory, or on disk; this is transparent to
       the OpenDevice caller.

       A full path name for the device name is legitimate.  For example
       "test:devs/fred.device".  This allows the use of custom devices
       without requiring the user to copy the device into the system's
       DEVS: directory.

   NOTES
       All calls to OpenDevice should have matching calls to CloseDevice!

       Devices on disk cannot be opened until after DOS has been
       started.

       As of V36 tasks can safely call OpenDevice, though DOS may open
       system requesters (e.g., asking the user to insert the Workbench
       disk if DEVS: is not online).  You must call this function from a
       DOS Process if you want to turn off DOS requesters.

   INPUTS
       devName - requested device name

       unitNumber - the unit number to open on that device.  The format of
               the unit number is device specific.  If the device does
               not have separate units, send a zero.

       iORequest - the I/O request block to be returned with
               appropriate fields initialized.

       flags - additional driver specific information.  This is sometimes
               used to request opening a device with exclusive access.

   RESULTS
       error - Returns a sign-extended copy of the io_Error field
               of the IORequest.  Zero if successful, else an error code
               is returned.

   BUGS
       AmigaDOS file names are not case sensitive, but Exec lists are.  If
       the library name is specified in a different case than it exists on
       disk, unexpected results may occur.

       Prior to V36, tasks could not make OpenDevice calls requiring disk
       access (since tasks are not allowed to make dos.library calls).
       Now OpenDevice is protected from tasks.

   SEE ALSO
       CloseDevice, DoIO, SendIO, CheckIO, AbortIO, WaitIO
```

```
exec.library/OpenLibrary                               exec.library/OpenLibrary

   NAME
       OpenLibrary -- gain access to a library

   SYNOPSIS
       library = OpenLibrary(libName, version)
       D0                    A1       D0

       struct Library *OpenLibrary(STRPTR, ULONG);

   FUNCTION
       This function returns a pointer to a library that was previously
       installed into the system.  If the requested library is exists, and
       if the library version is greater than or equal to the requested
       version, then the open will succeed.

       The device may exist in memory, or on disk; this is transparent to
       the OpenDevice caller.  Only Processes are allowed to call
       OpenLibrary (since OpenLibrary may in turn call dos.library).

       A full path name for the library name is legitimate.  For example
       "wp:libs/wp.library".  This allows the use of custom libraries
       without requiring the user to copy the library into the system's
       LIBS: directory.

   NOTES
       All calls to OpenLibrary should have matching calls to CloseLibrary!

       Libraries on disk cannot be opened until after DOS has been
       started.

       As of V36 tasks can safely call OpenLibrary, though DOS may open
       system requesters (e.g., asking the user to insert the Workbench
       disk if LIBS: is not online).  You must call this function from a
       DOS Process if you want to turn off DOS requesters.

   INPUTS
       libName - the name of the library to open

       version - the version of the library required.

   RESULTS
       library - a library pointer for a successful open, else zero

   BUGS
       AmigaDOS file names are not case sensitive, but Exec lists are. If
       the library name is specified in a different case than it exists on
       disk, unexpected results may occur.

       Prior to V36, tasks could not make OpenLibrary calls requiring disk
       access (since tasks are not allowed to make dos.library calls).
       Now OpenLibrary is protected from tasks.

       The version number of the resident tag in disk based library must
       match the version number of the library, or V36 may fail to load it.

   SEE ALSO
       CloseLibrary
```

```
exec.library/OpenResource                             exec.library/OpenResource

   NAME
       OpenResource -- gain access to a resource

   SYNOPSIS
       resource = OpenResource(resName)
       D0                      A1

       APTR OpenResource(STRPTR);

   FUNCTION
       This function returns a pointer to a resource that was previously
       installed into the system.

       There is no CloseResource() function.

   INPUTS
      resName - the name of the resource requested.

   RESULTS
       resource - if successful, a resource pointer, else NULL
```

```
exec.library/Permit                                    exec.library/Permit

    NAME
        Permit -- permit task rescheduling.

    SYNOPSIS
        Permit()

        void Permit(void);

    FUNCTION
        Allow other tasks to be scheduled to run by the dispatcher, after a
        matching Forbid() has been executed.

    RESULTS
        Other tasks will be rescheduled as they are ready to run. In order
        to restore normal task rescheduling, the programmer must execute
        exactly one call to Permit() for every call to Forbid().

    NOTE
        This call is guaranteed to preserve all registers.

    SEE ALSO
        Forbid, Disable, Enable
```

```
exec.library/Procure                                   exec.library/Procure

    NAME
        Procure -- bid for a message lock (semaphore)

    SYNOPSIS
        result = Procure(semaphore, bidMessage)
        D0                A0         A1

        BYTE Procure(struct Semaphore *, struct Message *);

    FUNCTION
        This function is used to obtain a message based semaphore lock.  If
        the lock is immediate, Procure() returns a true result, and the
        bidMessage is not used.   If the semaphore is already locked,
        Procure() returns false, and the task must wait for the bidMessage
        to arrive at its reply port.

        Straight "Semaphores" use the message system.  They are therefore
        queueable, and users may wait on several of them at the same time.
        This makes them more powerful than "Signal Semaphores"

    INPUT
        semaphore - a semaphore message port.  This port is used to queue
        all pending lockers.  This port should be initialized with the
        PA_IGNORE option, as the MP_SigTask field is used for a pointer to
        the current locker message (not a task). New semaphore ports must
        also have the SM_BIDS word initialized to -1.  If the semaphore is
        public, it should be named, its priority set, and the added with
        AddPort. Message port priority is often used for anti-deadlock
        locking conventions.

    RESULT
        result - true when the semaphore is free.  In such cases no waiting
        needs to be done.  If false, then the task should wait at its
        bidMessage reply port.

    BUGS
        Procure() and Vacate() do not have proven reliability.

    SEE ALSO
        Vacate()
```

exec.library

Page 77

```
exec.library/PutMsg                                        exec.library/PutMsg

   NAME
        PutMsg -- put a message to a message port

   SYNOPSIS
        PutMsg(port, message)
               A0    A1

        void PutMsg(struct MsgPort *, struct Message *);

   FUNCTION
        This function attaches a message to the end of a given message port.
        It provides a fast, non-copying message sending mechanism.

        Messages can be attached to only one port at a time.  The message
        body can be of any size or form.  Because messages are not copied,
        cooperating tasks share the same message memory.  The sender task
        must not recycle the message until it has been replied by the
        receiver.  Of course this depends on the message handling conventions
        setup by the involved tasks.  If the ReplyPort field is non-zero,
        when the message is replied by the receiver, it will be sent back to
        that port.

        Any one of the following actions can be set to occur when a message
        is put:

                1. no special action
                2. signal a given task (specified by MP_SIGTASK)
                3. cause a software interrupt (specified by MP_SIGTASK)

        The action is selected depending on the value found in the MP_FLAGS
        of the destination port.

   IMPLEMENTATION
        1.  Sets the LN_TYPE field to "NT_MESSAGE".
        2.  Attaches the message to the destination port.
        3.  Performs the specified arrival action at the destination.

   INPUT
        port - pointer to a message port
        message - pointer to a message

   SEE ALSO
        GetMsg, ReplyMsg, exec/ports.h
```

exec.library

Page 78

```
exec.library/RawDoFmt                                      exec.library/RawDoFmt

   NAME
        RawDoFmt -- format data into a character stream.

   SYNOPSIS
        RawDoFmt(FormatString, DataStream, PutChProc, PutChData);
                 a0            a1          a2         a3

        void RawDoFmt(STRPTR,APTR,void (*)(),APTR);

   FUNCTION
        perform "C"-language-like formatting of a data stream, outputting
        the result a character at a time.  Where % formatting commands are
        found in the FormatString, they will be replaced with the
        corresponding element in the DataStream.  %% must be used in the
        string if a % is desired in the output.

        Under V36, RawDoFmt() returns a pointer to the end of the DataStream
        (The next argument that would have been processed).  This allows
        multiple formatting passes to be made using the same data.

   INPUTS
        FormatString - a "C"-language-like NULL terminated format string,
        with the following supported % options:

         %[flags][width.limit][length]type

        flags  - only one allowed. '-' specifies left justification.
        width  - field width.  If the first character is a '0', the
                 field will be padded with leading 0's.
          .    - must follow the field width, if specified
        limit  - maximum number of characters to output from a string.
                 (only valid for %s).
        length - size of input data defaults to WORD for types d, x,
                 and c, 'l' changes this to long (32-bit).
        type   - supported types are:
                        b - BSTR, data is 32-bit BPTR to byte count followed
                            by a byte string, or NULL terminated byte string.
                            A NULL BPTR is treated as an empty string.
                            (Added in V36 exec)
                        d - decimal
                        x - hexadecimal
                        s - string, a 32-bit pointer to a NULL terminated
                            byte string.  In V36, a NULL pointer is treated
                            as an empty string
                        c - character

        DataStream - a stream of data that is interpreted according to
                 the format string.  Often this is a pointer into
                 the task's stack.
        PutChProc  - the procedure to call with each character to be
                 output, called as:

        PutChProc(Char,  PutChData);
                  D0-0:8 A3

                the procedure is called with a NULL Char at the end of
                the format string.

        PutChData - a value that is passed through to the PutChProc
                procedure.  This is untouched by RawDoFmt, and may be
                modified by the PutChProc.

   EXAMPLE
        ;
```

```
; Simple version of the C "sprintf" function.  Assumes C-style
; stack-based function conventions.
;
;   long eyecount;
;   eyecount=2;
;   sprintf(string,"%s have %ld eyes.","Fish",eyecount);
;
; would produce "Fish have 2 eyes." in the string buffer.
;
        XDEF _sprintf
        XREF _AbsExecBase
        XREF _LVORawDoFmt
_sprintf:       ; ( ostring, format, {values} )
        movem.l a2/a3/a6,-(sp)

        move.l  4*4(sp),a3       ;Get the output string pointer
        move.l  5*4(sp),a0       ;Get the FormatString pointer
        lea.l   6*4(sp),a1       ;Get the pointer to the DataStream
        lea.l   stuffChar(pc),a2
        move.l  _AbsExecBase,a6
        jsr     _LVORawDoFmt(a6)

        movem.l (sp)+,a2/a3/a6
        rts

;------ PutChProc function used by RawDoFmt -----------
stuffChar:
        move.b  d0,(a3)+        ;Put data to output string
        rts
```

WARNING

This Amiga ROM function formats word values in the data stream. If your compiler defaults to longs, you must add an "l" to your % specifications. This can get strange for characters, which might look like "%lc".

SEE ALSO

Documentation on the C language "printf" call in any C language reference book.

exec.library/ReleaseSemaphore   exec.library/ReleaseSemaphore

NAME

ReleaseSemaphore -- make signal semaphore available to others

SYNOPSIS

```
ReleaseSemaphore(signalSemaphore)
                 A0

void ReleaseSemaphore(struct SignalSemaphore *);
```

FUNCTION

ReleaseSemaphore() is the inverse of ObtainSemaphore(). It makes the semaphore lockable to other users. If tasks are waiting for the semaphore and this this task is done with the semaphore then the next waiting task is signalled.

Each ObtainSemaphore() call must be balanced by exactly one ReleaseSemaphore() call. This is because there is a nesting count maintained in the semaphore of the number of times that the current task has locked the semaphore. The semaphore is not released to other tasks until the number of releases matches the number of obtains.

Needless to say, havoc breaks out if the task releases more times than it has obtained.

INPUT

signalSemaphore -- an initialized signal semaphore structure

NOTE

This call is guaranteed to preserve all registers.

SEE ALSO

InitSemaphore(), ObtainSemaphore(), ObtainSemaphoreShared()

```
exec.library/ReleaseSemaphoreList          exec.library/ReleaseSemaphoreList

   NAME
        ReleaseSemaphoreList -- make a list of semaphores available

   SYNOPSIS
        ReleaseSemaphoreList(list)
                             A0

        void ReleaseSemaphoreList(struct List *);

   FUNCTION
        ReleaseSemaphoreList() is the inverse of ObtainSemaphoreList(). It
        releases each element in the semaphore list.

        Needless to say, havoc breaks out if the task releases more times
        than it has obtained.

   INPUT
       list -- a list of signal semaphores

   SEE ALSO
        ObtainSemaphoreList()
```

```
exec.library/RemDevice                                exec.library/RemDevice

   NAME
        RemDevice -- remove a device from the system

   SYNOPSIS
        RemDevice(device)
                  A1

        void RemDevice(struct Device *);

   FUNCTION
        This function calls the device's EXPUNGE vector, which requests
        that a device delete itself.  The device may refuse to do this if
        it is busy or currently open. This is not typically called by user
        code.

        There are certain, limited circumstances where it may be
        appropriate to attempt to specifically flush a certain device.
        Example:

         /* Attempts to flush the named device out of memory. */
         #include <exec/types.h>
         #include <exec/execbase.h>

         void FlushDevice(name)
         STRPTR name;
         {
         struct Device *result;

            Forbid();
            if(result=(struct Device *)FindName(&SysBase->DeviceList,name))
                RemDevice(result);
            Permit();
         }

   INPUTS
        device - pointer to a device node

   SEE ALSO
        AddLibrary
```

```
exec.library/RemHead                                     exec.library/RemHead

   NAME
        RemHead -- remove the head node from a list

   SYNOPSIS
        node = RemHead(list)
        D0             A0

        struct Node *RemHead(struct List *);

   FUNCTION
        Get a pointer to the head node and remove it from the list.
        Assembly programmers may prefer to use the REMHEAD macro from
        "exec/lists.i".

   WARNING
        This function does not arbitrate for access to the list.  The
        calling task must be the owner of the involved list.

   INPUTS
        list - a pointer to the target list header

   RESULT
        node - the node removed or zero when empty list

   SEE ALSO
        AddHead, AddTail, Enqueue, Insert, Remove, RemTail
```

```
exec.library/RemIntServer                           exec.library/RemIntServer

   NAME
        RemIntServer -- remove an interrupt server from a server chain

   SYNOPSIS
        RemIntServer(intNum, interrupt)
                     D0      A1

        void RemIntServer(ULONG,struct Interrupt *);

   FUNCTION
        This function removes an interrupt server node from the given
        server chain.

        If this server was the last one on this chain, interrupts for this
        chain are disabled.

   INPUTS
        intNum - the Portia interrupt bit (0..14)
        interrupt - pointer to an interrupt server node

   BUGS
        Before V36 Kickstart, the feature that disables the interrupt
        would not function.  For most server chains this does not
        cause a problem.

   SEE ALSO
        AddIntServer, hardware/intbits.h
```

exec.library Page 85

```
exec.library/RemLibrary                                   exec.library/RemLibrary

   NAME
        RemLibrary -- remove a library from the system

   SYNOPSIS
        RemLibrary(library)
                   A1

        void RemLibrary(struct Library *);

   FUNCTION
        This function calls the library's EXPUNGE vector, which requests
        that a library delete itself.  The library may refuse to do this if
        it is busy or currently open. This is not typically called by user
        code.

        There are certain, limited circumstances where it may be
        appropriate to attempt to specifically flush a certain Library.
        Example:

         /* Attempts to flush the named library out of memory. */
         #include <exec/types.h>
         #include <exec/execbase.h>

         void FlushLibrary(name)
         STRPTR name;
         {
         struct Library *result;

            Forbid();
            if(result=(struct Library *)FindName(&SysBase->LibList,name))
                RemLibrary(result);
            Permit();
         }

   INPUTS
        library - pointer to a library node structure
```

exec.library Page 86

```
exec.library/Remove                                           exec.library/Remove

   NAME
        Remove -- remove a node from a list

   SYNOPSIS
        Remove(node)
               A1

        void Remove(struct Node *);

   FUNCTION
        Unlink a node from whatever list it is in.  Nodes that are not part
        of a list must not be passed to this funcion!  Assembly programmers
        may prefer to use the REMOVE macro from "exec/lists.i".

   WARNING
        This function does not arbitrate for access to the list.  The
        calling task must be the owner of the involved list.

   INPUTS
        node - the node to remove

   SEE ALSO
        AddHead, AddTail, Enqueue, Insert, RemHead, RemTail
```

```
exec.library/RemPort                                       exec.library/RemPort

   NAME
        RemPort -- remove a message port from the system

   SYNOPSIS
        RemPort(port)
                A1

        void RemPort(struct MsgPort *);

   FUNCTION
        This function removes a message port structure from the system's
        message port list.  Subsequent attempts to rendezvous by name with
        this port will fail.

   INPUTS
        port - pointer to a message port

   SEE ALSO
        AddPort, FindPort
```

```
exec.library/RemResource                               exec.library/RemResource

   NAME
        RemResource -- remove a resource from the system

   SYNOPSIS
        RemResource(resource)
                    A1

        void RemResource(APTR);

   FUNCTION
        This function removes an existing resource from the system resource
        list.  There must be no outstanding users of the resource.

   INPUTS
        resource - pointer to a resource node

   SEE ALSO
        AddResource
```

exec.library Page 89

```
exec.library/RemSemaphore                              exec.library/RemSemaphore

   NAME
        RemSemaphore -- remove a signal semaphore from the system

   SYNOPSIS
        RemSemaphore(signalSemaphore)
                     A1

        void RemSemaphore(struct SignalSemaphore *);

   FUNCTION
        This function removes a signal semaphore structure from the
        system's signal semaphore list.  Subsequent attempts to
        rendezvous by name with this semaphore will fail.

   INPUTS
       signalSemaphore -- an initialized signal semaphore structure

   SEE ALSO
        AddSemaphore, FindSemaphore
```

exec.library Page 90

```
exec.library/RemTail                                        exec.library/RemTail

   NAME
        RemTail -- remove the tail node from a list

   SYNOPSIS
        node = RemTail(list)
        D0             A0

        struct Node *RemTail(struct List *);

   FUNCTION
        Remove the last node from a list, and return a pointer to it. If
        the list is empty, return zero. Assembly programmers may prefer to
        use the REMTAIL macro from "exec/lists.i".

   WARNING
        This function does not arbitrate for access to the list.  The
        calling task must be the owner of the involved list.

   INPUTS
        list - a pointer to the target list header

   RESULT
        node - the node removed or zero when empty list

   SEE ALSO
        AddHead, AddTail, Enqueue, Insert, Remove, RemHead, RemTail
```

```
exec.library/RemTask                                     exec.library/RemTask

   NAME
       RemTask -- remove a task from the system

   SYNOPSIS
       RemTask(task)
               A1

       void RemTask(struct Task *);

   FUNCTION
       This function removes a task from the system.  Deallocation of
       resources should have been performed prior to calling this
       function.  Removing some other task is very dangerous.  Generally
       is is best to arrange for tasks to call RemTask(0L) on themselves.

       RemTask will automagically free any memory lists attached to the
       task's TC_MEMENTRY list.

   INPUTS
       task - pointer to the task node representing the task to be
              removed.  A zero value indicates self removal, and will
              cause the next ready task to begin execution.

   BUGS
       Before V36 if RemTask() was called on a task other than the current
       task, and that task was created with amiga.lib/CreateTask, there was
       a slight chance of a crash.  The problem can be hidden by bracketing
       RemTask() with Forbid()/Permit().

   SEE ALSO
       AddTask, exec/AllocEntry, amiga.lib/DeleteTask
```

```
exec.library/ReplyMsg                                   exec.library/ReplyMsg

   NAME
       ReplyMsg -- put a message to its reply port

   SYNOPSIS
       ReplyMsg(message)
                A1

       void ReplyMsg(struct Message *);

   FUNCTION
       This function sends a message to its reply port.  This is usually
       done when the receiver of a message has finished and wants to
       return it to the sender (so that it can be re-used or deallocated,
       whatever).

       This call may be made from interrupts.

   INPUT
       message - a pointer to the message

   IMPLEMENTATION
       1> Places "NT_REPLYMSG" into LN_TYPE.
       2> Puts the message to the port specified by MN_REPLYPORT
          If there is no replyport, sets LN_TYPE to "NT_FREEMSG" (use this
          feature only with extreeme care).

   SEE ALSO
       GetMsg, PutMsg, exec/ports.h
```

exec.library Page 93

```
exec.library/SendIO                                    exec.library/SendIO

   NAME
        SendIO -- initiate an I/O command

   SYNOPSIS
        SendIO(iORequest)
               A1

        void SendIO(struct IORequest *);

   FUNCTION
        This function requests the device driver start processing the given
        I/O request.  The device will return control without waiting for
        the I/O to complete.

        The io_Flags field of the IORequest will be set to zero before the
        request is sent.  See BeginIO() for more details.

   INPUTS
        iORequest - pointer to an I/O request, or a device specific
                    extended IORequest.

   SEE ALSO
        DoIO, CheckIO, WaitIO, AbortIO
```

exec.library Page 94

```
exec.library/SetExcept                                  exec.library/SetExcept

   NAME
        SetExcept -- define certain signals to cause exceptions

   SYNOPSIS
        oldSignals = SetExcept(newSignals, signalMask)
        D0                     D0          D1

        ULONG SetExcept(ULONG,ULONG);

   FUNCTION
        This function defines which of the task's signals will cause a
        private task exception.  When any of the signals occurs the task's
        exception handler will be dispatched.  If the signal occurred prior
        to calling SetExcept, the exception will happen immediately.

        The user function pointed to by the task's tc_ExceptCode gets
        called as:

            newExcptSet = <exceptCode>(signals, exceptData),SysBase
            D0                         D0       A1          A6

            signals - The set of signals that caused this exception.  These
                Signals have been disabled from the current set of signals
                that can cause an exception.

            exceptData - A copy of the task structure tc_ExceptData field.

            newExcptSet - The set of signals in NewExceptSet will be re-
                enabled for exception generation.  Usually this will be the
                same as the Signals that caused the exception.

   INPUTS
        newSignals - the new values for the signals specified in
                signalMask.
        signalMask - the set of signals to be effected

   RESULTS
        oldSignals - the prior exception signals

   EXAMPLE
        Get the current state of all exception signals:
            SetExcept(0,0)
        Change a few exception signals:
            SetExcept($1374,$1074)

   SEE ALSO
        Signal, SetSignal
```

exec.library Page 95

```
exec.library/SetFunction                              exec.library/SetFunction

   NAME
       SetFunction -- change a function vector in a library

   SYNOPSIS
       oldFunc = SetFunction(library, funcOffset, funcEntry)
       D0                    A1       A0.W        D0

       APTR SetFunction(struct Library *,LONG,APTR);

   FUNCTION
       SetFunction is a functional way of changing where vectors in a
       library point.  They are changed in such a way that the
       checksumming process will never falsely declare a library to be
       invalid.

   NOTE
       SetFunction cannot be used on non-standard libraries like
       dos.library.  Here you must manually Forbid(), preserve all 6
       original bytes, set the new vector, SumLibrary(), then Permit().

   INPUTS
       library    - a pointer to the library to be changed
       funcOffset - the offset of the function to be replaced
       funcEntry  - pointer to new function

   RESULTS
       oldFunc    - pointer to the old function that was just replaced
```

exec.library Page 96

```
exec.library/SetIntVector                            exec.library/SetIntVector

   NAME
       SetIntVector -- set a new handler for a system interrupt vector

   SYNOPSIS
       oldInterrupt = SetIntVector(intNumber, interrupt)
       D0                          D0         A1

       struct Interrupt *SetIntVector(ULONG, struct Interrupt *);

   FUNCTION
       This function provides a mechanism for setting the system interrupt
       vectors.  These are non-sharable; setting a new interrupt handler
       disconnects the old one.  Installed handlers are responsible for
       processing, enabling and clearing the interrupt.  Note that interrupts
       may have been left in any state by the previous code.

       The IS_CODE and IS_DATA pointers of the Interrupt structure will
       be copied into a private place by Exec.  A pointer to the previously
       installed Interrupt structure is returned.

       When the system calls the specified interrupt code, the registers are
       setup as follows:

           D0 - scratch
           D1 - scratch (on entry: active portia
                         interrupts -> equals INTENA & INTREQ)

           A0 - scratch (on entry: pointer to base of custom chips
                         for fast indexing)
           A1 - scratch (on entry: Interrupt's IS_DATA pointer)

           A5 - jump vector register (scratch on call)
           A6 - Exec library base pointer (scratch on call)

           all other registers must be preserved

   INPUTS
       intNum - the Portia interrupt bit number (0..14).  Only non-chained
                interrupts should be set.  Use AddIntServer() for server
                chains.
       interrupt - a pointer to an Interrupt structure containing the
                handler's entry point and data segment pointer.  A NULL
                interrupt pointer will remove the current interrupt and
                set illegal values for IS_CODE and IS_DATA.

                By convention, the LN_NAME of the interrupt structure must
                point a descriptive string so that other users may
                identify who currently has control of the interrupt.

   RESULT
       A pointer to the prior interrupt structure which had control
       of this interrupt.

   SEE ALSO
       AddIntServer(),exec/interrupts.i,exec/hardware.i
```

exec.library Page 97

```
exec.library/SetSignal                                exec.library/SetSignal

   NAME
        SetSignal -- define the state of this task's signals

   SYNOPSIS
        oldSignals = SetSignal(newSignals, signalMask)
        D0                     D0          D1

        ULONG SetSignal(ULONG,ULONG);

   FUNCTION
        This function can query or modify the state of the current task's
        received signal mask.  Setting the state of signals is considered
        dangerous.  Reading the state of signals is safe.

   INPUTS
        newSignals - the new values for the signals specified in
                     signalSet.
        signalMask - the set of signals to be affected.

   RESULTS
        oldSignals - the prior values for all signals

   EXAMPLES
        Get the current state of all signals:
            SetSignal(0L,0L);
        Clear all signals:
            SetSignal(0L,0xFFFFFFFFL);
        Clear the CTRL-C signal:
            SetSignal(0L,SIGBREAKF_CTRL_C);


        Check if the CTRL-C signal was pressed:

            #include <libraries/dos.h>

            if(SetSignal(0L,0L) & SIGBREAKF_CTRL_C) /* Hit since last check? */
                {
                SetSignal(0L,SIGBREAKF_CTRL_C); /* Clear old status */
                printf("CTRL-C pressed!\n");
                }

   SEE ALSO
        Signal, Wait
```

exec.library Page 98

```
exec.library/SetSR                                          exec.library/SetSR

   NAME
        SetSR -- get and/or set processor status register

   SYNOPSIS
        oldSR = SetSR(newSR, mask)
        D0            D0     D1

        ULONG SetSR(ULONG, ULONG);

   FUNCTION
        This function provides a means of modifying the CPU status register
        in a "safe" way (well, how safe can a function like this be
        anyway?).  This function will only affect the status register bits
        specified in the mask parameter.  The prior content of the entire
        status register is returned.

   INPUTS
        newSR - new values for bits specified in the mask.
            All other bits are not effected.
        mask - bits to be changed

   RESULTS
        oldSR - the entire status register before new bits

   EXAMPLES
        To get the current SR:
            currentSR = SetSR(0,0);
        To change the processor interrupt level to 3:
            oldSR = SetSR($0300,$0700);
        Set processor interrupts back to prior level:
            SetSR(oldSR,$0700);
```

```
exec.library/SetTaskPri                                 exec.library/SetTaskPri

   NAME
       SetTaskPri -- get and set the priority of a task

   SYNOPSIS
       oldPriority = SetTaskPri(task, priority)
       D0-0:8                   A1    D0-0:8

       BYTE SetTaskPri(struct Task *,LONG);

   FUNCTION
       This function changes the priority of a task regardless of its
       state.  The old priority of the task is returned.  A reschedule is
       performed, and a context switch may result.

       To change the priority of the currently running task, pass the
       result of FindTask(0); as the task pointer.

   INPUTS
       task - task to be affected
       priority - the new priority for the task

   RESULT
       oldPriority - the tasks previous priority
```

```
exec.library/Signal                                         exec.library/Signal

   NAME
       Signal -- signal a task

   SYNOPSIS
       Signal(task, signals)
              A1    D0

       void Signal(struct Task *,ULONG);

   FUNCTION
       This function signals a task with the given signals.  If the task
       is currently waiting for one or more of these signals, it will be
       made ready and a reschedule will occur. If the task is not waiting
       for any of these signals, the signals will be posted to the task
       for possible later use. A signal may be sent to a task regardless
       of whether its running, ready, or waiting.

       This function is considered "low level".  Its main purpose is to
       support multiple higher level functions like PutMsg.

       This function is safe to call from interrupts.

   INPUT
       task - the task to be signalled
       signals - the signals to be sent

   SEE ALSO
       Wait, SetSignal
```

exec.library/SumKickData   exec.library/SumKickData

NAME

SumKickData -- compute the checksum for the Kickstart delta list

SYNOPSIS

```
checksum = SumKickData()
D0

ULONG SumKickData(void);
```

FUNCTION

The Amiga system has some ROM (or Kickstart) resident code that provides the basic functions for the machine. This code is unchangeable by the system software. This function is part of a support system to modify parts of the ROM.

The ROM code is linked together at run time via ROMTags (also known as Resident structures, defined in exec/resident.h). These tags tell Exec's low level boot code what subsystems exist in which regions of memory. The current list of ROMTags is contained in the ResModules field of ExecBase. By default this list contains any ROMTags found in the address ranges $F80000-$FFFFFF and $F00000-$F7FFFF.

There is also a facility to selectively add or replace modules to the ROMTag list. These modules can exist in RAM, and the memory they occupy will be deleted from the memory free list during the boot process. SumKickData() plays an important role in this run-time modification of the ROMTag array.

Three variables in ExecBase are used in changing the ROMTag array: KickMemPtr, KickTagPtr, and KickCheckSum. KickMemPtr points to a linked list of MemEntry structures. The memory that these MemEntry structures reference will be allocated (via AllocAbs) at boot time. The MemEntry structure itself must also be in the list.

KickTagPtr points to a long-word array of the same format as the ResModules array. The array has a series of pointers to ROMTag structures. The array is either NULL terminated, or will have an entry with the most significant bit (bit 31) set. The most significant bit being set says that this is a link to another long-word array of ROMTag entries. This new array's address can be found by clearing bit 31.

KickCheckSum has the result of SumKickData(). It is the checksum of both the KickMemPtr structure and the KickTagPtr arrays. If the checksum does not compute correctly then both KickMemPtr and KickTagPtr will be ignored.

If all the memory referenced by KickMemPtr can't be allocated then KickTagPtr will be ignored.

There is one more important caveat about adding ROMTags. All this ROMTag magic is run very early on in the system -- before expansion memory is added to the system. Therefore any memory in this additional ROMTag area must be addressable at this time. This means that your ROMTag code, MemEntry structures, and resident arrays cannot be in expansion memory. There are two regions of memory that are acceptable: one is chip memory, and the other is "Ranger" memory (memory in the range between $C00000-$D80000).

Remember that changing an existing ROMTag entry falls into the "heavy magic" category -- be very careful when doing it. The odd are that you will blow yourself out of the water.

NOTE

SumKickData was introduced in the 1.2 release

RESULT

Value to be stuffed into ExecBase->KickCheckSum.

SEE ALSO

InitResident, FindResident

```
exec.library/SumLibrary                           exec.library/SumLibrary

   NAME
        SumLibrary -- compute and check the checksum on a library

   SYNOPSIS
        SumLibrary(library)
                   A1

        void SumLibrary(struct Library *);

   FUNCTION
        SumLibrary computes a new checksum on a library.  It can also be
        used to check an old checksum.  If an old checksum does not match,
        and the library has not been marked as changed, then the system
        will call Alert().

        This call could also be periodically made by some future
        system-checking task.

   INPUTS
        library - a pointer to the library to be changed

   NOTE
        An alert will occur if the checksum fails.

   SEE ALSO
        SetFunction
```

```
exec.library/SuperState                           exec.library/SuperState

   NAME
        SuperState -- enter supervisor state with user stack

   SYNOPSIS
        oldSysStack = SuperState()
        D0

        APTR SuperState(void);

   FUNCTION
        Enter supervisor mode while running on the user's stack. The user
        still has access to user stack variables.  Be careful though, the
        user stack must be large enough to accommodate space for all
        interrupt data -- this includes all possible nesting of interrupts.
        This function does nothing when called from supervisor state.

   RESULTS
        oldSysStack - system stack pointer; save this.  It will come in
                      handy when you return to user state.  If the system
                      is already in supervisor mode, oldSysStack is zero.

   SEE ALSO
        UserState/Supervisor
```

```
exec.library/Supervisor                               exec.library/Supervisor

   NAME
       Supervisor -- trap to a short supervisor mode function

   SYNOPSIS
       result = Supervisor(userFunc)
       Rx                  A5

       ULONG Supervisor(void *);

   FUNCTION
       Allow a normal user-mode program to execute a short assembly language
       function in the supervisor mode of the processor.  Supervisor() does
       not modify or save registers; the user function has full access to the
       register set.   All rules that apply to interrupt code must be
       followed.  In addition, no system calls are permitted.  The function
       must end with an RTE instruction.

   EXAMPLE
               ;Obtain the Exception Vector base.  68010 or greater only!
               MOVECtrap:      movec.l VBR,d0  ;$4e7a,$0801
                               rte

   INPUTS
       userFunc - A pointer to a short assembly language function ending
                  in RTE.  The function has full access to the register set.

   RESULTS
       result   - Whatever values the userFunc left in the registers.

   SEE ALSO
       SuperState,UserState
```

```
exec.library/TypeOfMem                                 exec.library/TypeOfMem

   NAME
       TypeOfMem -- determine attributes of a given memory address

   SYNOPSIS
       attributes = TypeOfMem(address)
       D0                     A1

       ULONG TypeOfMem(void *);

   FUNCTION
       Given a RAM memory address, search the system memory lists and
       return its memory attributes.  The memory attributes are similar to
       those specified when the memory was first allocated: (eg. MEMF_CHIP
       and MEMF_FAST).

       This function is usually used to determine if a particular block of
       memory is within CHIP space.

       If the address is not in known-space, a zero will be returned.
       (Anything that is not RAM, like the ROM or expansion area, will
       return zero.  Also the first few bytes of a memory area are used up
       by the MemHeader.)

   INPUT
       address - a memory address

   RESULT
       attributes - a long word of memory attribute flags.
       If the address is not in known RAM, zero is returned.

   SEE ALSO
       AllocMem()
```

exec.library Page 107

```
exec.library/UserState                                   exec.library/UserState

   NAME
        UserState -- return to user state with user stack

   SYNOPSIS
        UserState(sysStack)
                  D0

        void UserState(APTR);

   FUNCTION
        Return to user state with user stack, from supervisor state with
        user stack.  This function is normally used in conjunction with the
        SuperState function above.

        This function must not be called from the user state.

   INPUT
        sysStack - supervisor stack pointer

   BUGS
        This function is broken in V33/34 Kickstart.  Fixed in V1.31 setpatch.

   SEE ALSO
        SuperState/Supervisor
```

---

exec.library Page 108

```
exec.library/Vacate                                          exec.library/Vacate

   NAME
        Vacate -- release a message lock (semaphore)

   SYNOPSIS
        Vacate(semaphore)
               A0

        void Vacate(struct Semaphore *);

   FUNCTION
        This function releases a previously locked semaphore (see
        the Procure() function).
        If another task is waiting for the semaphore, its bidMessage
        will be sent to its reply port.

   INPUT
        semaphore - the semaport message port representing the
        semaphore to be freed.

   BUGS
        Procure() and Vacate() do not have proven reliability.

   SEE ALSO
        Procure
```

```
exec.library/Wait                                          exec.library/Wait

   NAME
        Wait -- wait for one or more signals

   SYNOPSIS
        signals = Wait(signalSet)
        D0             D0

        ULONG Wait(ULONG);

   FUNCTION
        This function will cause the current task to suspend waiting for
        one or more signals.  When one or more of the specified signals
        occurs, the task will return to the ready state, and those signals
        will be cleared.

        If a signal occurred prior to calling Wait(), the wait condition will
        be immediately satisfied, and the task will continue to run without
        delay.

   CAUTION
        This function cannot be called while in supervisor mode or
        interrupts!  This function will break the action of a Forbid() or
        Disable() call.

   INPUT
        signalSet - The set of signals for which to wait.
                    Each bit represents a particular signal.

   RESULTS
        signals - the set of signals that were active
```

```
exec.library/WaitIO                                      exec.library/WaitIO

   NAME
        WaitIO -- wait for completion of an I/O request

   SYNOPSIS
        error = WaitIO(iORequest)
        D0             A1

        BYTE WaitIO(struct IORequest *);

   FUNCTION
        This function waits for the specified I/O request to complete, then
        removes it from the replyport.  If the I/O has already completed,
        this function will return immediately.

        This function should be used with care, as it does not return until
        the I/O request completes; if the I/O never completes, this
        function will never return, and your task will hang.  If this
        situation is a possibility, it is safer to use the Wait() function.
        Wait() will return return when any of a specified set of signal is
        received.  This is how I/O timeouts can be properly handled.

   WARNING
        If this IORequest was "Quick" or otherwise finished BEFORE this
        call, this function drops though immediately, with no call to
        Wait().  A side effect is that the signal bit related the port may
        remain set.  Expect this.

        When removing a known complete IORequest from a port, WaitIO() is the
        preferred method.  A simple Remove() would require a Disable/Enable
        pair!

   INPUTS
        iORequest - pointer to an I/O request block

   RESULTS
        error - zero if successful, else an error is returned
                (a sign extended copy of io_Error).

   SEE ALSO
        DoIO, SendIO, CheckIO, AbortIO
```

```
exec.library/WaitPort                                   exec.library/WaitPort

   NAME
        WaitPort -- wait for a given port to be non-empty

   SYNOPSIS
        message = WaitPort(port)
        D0                 A0

        struct Message *WaitPort(struct MsgPort *);

   FUNCTION
        This function waits for the given port to become non-empty.  If
        necessary, the Wait() function will be called to wait for the port
        signal.  If a message is already present at the port, this function
        will return immediately.  The return value is always a pointer to
        the first message queued (but it is not removed from the queue).

   CAUTION
        More than one message may be at the port when this returns.  It is
        proper to call the GetMsg() function in a loop until all messages
        have been handled, then wait for more to arrive.

        To wait for more than one port, combine the signal bits from each
        port into one call to the Wait() function, then use a GetMsg() loop
        to collect any and all messages.  It is possible to get a signal
        for a port WITHOUT a message showing up.  Plan for this.

   INPUT
        port - a pointer to the message port

   RETURN
        message - a pointer to the first available message

   SEE ALSO
        GetMsg
```

expansion.library

TABLE OF CONTENTS

expansion.library/AddBootNode
expansion.library/AddConfigDev
expansion.library/AddDosNode
expansion.library/AllocConfigDev
expansion.library/AllocExpansionMem
expansion.library/ConfigBoard
expansion.library/FindConfigDev
expansion.library/FreeConfigDev
expansion.library/FreeExpansionMem
expansion.library/GetCurrentBinding
expansion.library/MakeDosNode
expansion.library/ObtainConfigBinding
expansion.library/ReadExpansionByte
expansion.library/ReadExpansionRom
expansion.library/ReleaseConfigBinding
expansion.library/RemConfigDev
expansion.library/SetCurrentBinding
expansion.library/WriteExpansionByte

expansion.library

```
expansion.library/AddBootNode                    expansion.library/AddBootNode

   NAME
        AddBootNode -- Add a BOOTNODE to the system (V36)

   SYNOPSIS
        ok = AddBootNode( bootPri, flags, deviceNode, configDev )
        D0                D0       D1     A0          A1

        BOOL AddBootNode( BYTE,ULONG,struct DeviceNode *,struct ConfigDev * );

   FUNCTION
        This function will do one of two things:

                1> If dos is running, add a new disk type device immediatly.
                2> If dos is not yet running, save information for later
                   use by the system.

        This function works exactly like the AddDosNode() function, but
        allows an additional specification.  Autoboot from an expansion card
        before DOS is running requires the card's ConfigDev structure.

        Pass a NULL ConfigDev pointer to create a non-bootable node.

   NOTE
        This function eliminates the need to manually Enqueue a BOOTNODE
        onto an expansion.library list.  Be sure V36 expansion.library is
        available before calling this function!

   SEE ALSO
        AddDosNode
```

```
expansion.library/AddConfigDev                 expansion.library/AddConfigDev

   NAME
        AddConfigDev - add a new ConfigDev structure to the system

   SYNOPSIS
        AddConfigDev( configDev )
                      A0

   FUNCTION
        (Not typically called by user code)

        This routine adds the specified ConfigDev structure to the
        list of Configuration Devices in the system.

   INPUTS
        configDev - a valid ConfigDev structure.

   RESULTS

   EXCEPTIONS

   SEE ALSO
        RemConfigDev

   BUGS
```

```
expansion.library/AddDosNode                     expansion.library/AddDosNode

   NAME
        AddDosNode -- mount a disk to the system

   SYNOPSIS
        ok = AddDosNode( bootPri, flags, deviceNode )
        D0               D0       D1     A0

        BOOL AddDosNode( BYTE,ULONG,struct DeviceNode *);

   FUNCTION
        This routine makes sure that your disk device (or a device
        that wants to be treated as if it was a disk...) will be
        entered into the system.  If the dos is already up and
        running, then it will be entered immediately.  If the dos
        has not yet been run then the data will be recorded, and the
        dos will get it later.

        We try and boot off of each device in turn, based on priority.
        Floppies have a hard-coded priority.

        There is only one additional piece of magic done by AddDosNode.
        If there is no executable code specified in the deviceNode
        structure (e.g. dn_SegList, dn_Handler, and dn_Task are all
        null) then the standard dos file handler is used for your
        device.

        Documentation note: a "task" as used here is a dos-task, not
        an exec-task.  A dos-task, in the strictest sense, is the
        address of an exec-style message port.  In general, it is
        a pointer to a process's pr_MsgPort field (e.g. a constant
        number of bytes after an exec port).

   INPUTS
        bootPri -- a BYTE quantity with the boot priority for this disk.
            This priority is only for which disks should be looked at:
            the actual disk booted from will be the first disk with
            a valid boot block.  If no disk is found then the "bootme"
            hand will come up and the bootstrap code will wait for
            a floppy to be inserted.  Recommend priority assignments are:

                +5   -- unit zero for the floppy disk.  The floppy should
                        always be highest priority to allow the user to
                        abort out of a hard disk boot.
                 0   -- the run of the mill hard disk
                -5   -- a "network" disk (local disks should take priority).
                -128 -- don't even bother to boot from this device.

        flags -- additional flag bits for the call:
            ADNF_STARTPROC (bit 0) -- start a handler process immediately.
                Normally the process is started only when the device node
                is first referenced.  This bit is meaningless if you
                have already specified a handler process (non-null dn_Task).

        deviceNode -- a legal DOS device node, properly initialized.
                Typically this will be the result of a MakeDosNode().
                Special cases may require a custom-built device node.

   RESULTS
        ok - non-zero everything went ok, zero if we ran out of memory
            or some other weirdness happened.

   EXAMPLE
        /* enter a bootable disk into the system.  Start a file handler
        ** process immediately.
```

```
     */
     if(  AddDosNode( 0, ADNF_STARTPROC, MakeDosNode( paramPacket ) )  )
             ...AddDosNode ok...

 BUGS
     Before V36 Kickstart, no function existed to add BOOTNODES.
     If an older expansion.library is in use, driver code will need
     to manually construct a BootNode and Enqueue() it to eb_Mountlist.

 SEE ALSO
     MakeDosNode, AddBootNode
```

```
expansion.library/AllocConfigDev             expansion.library/AllocConfigDev

   NAME
        AllocConfigDev - allocate a ConfigDev structure

   SYNOPSIS
        configDev = AllocConfigDev()
        D0

   FUNCTION
        This routine returns the address of a ConfigDev structure.
        It is provided so new fields can be added to the structure
        without breaking old, existing code.  The structure is cleared
        when it is returned to the user.

   INPUTS

   RESULTS
        configDev - either a valid ConfigDev structure or NULL.

   EXCEPTIONS

   SEE ALSO
        FreeConfigDev

   BUGS
```

```
expansion.library/AllocExpansionMem       expansion.library/AllocExpansionMem

   NAME
        AllocExpansionMem - allocate expansion memory

   SYNOPSIS
        startSlot = AllocExpansionMem( numSlots, slotOffset )
        D0                             D0        D1

   FUNCTION
        (Not typically called by user code)

        This function allocates numslots of expansion space (each slot
        is E_SLOTSIZE bytes).  It returns the slot number of the
        start of the expansion memory.  The EC_MEMADDR macro may be
        used to convert this to a memory address.

        Boards that fit the expansion architecture have alignment
        rules.  Normally a board must be on a binary boundary of its
        size.  Four and Eight megabyte boards have special rules.
        User defined boards might have other special rules.

        If AllocExpansionMem() succeeds, the startSlot will satisfy
        the following equation:

                (startSlot - slotOffset) MOD slotAlign = 0

   INPUTS
        numSlots - the number of slots required.
        slotOffset - an offset from that boundary for startSlot.

   RESULTS
        startSlot - the slot number that was allocated, or -1 for error.

   EXAMPLES

                AllocExpansionMem( 2, 0 )

        Tries to allocate 2 slots on a two slot boundary.

                AllocExpansionMem( 64, 32 )

        This is the allocation rule for 4 meg boards.  It allocates
        4 megabytes (64 slots) on an odd 2 meg boundary.

   EXCEPTIONS

   SEE ALSO
        FreeExpansionMem

   BUGS
```

```
expansion.library/ConfigBoard                     expansion.library/ConfigBoard

   NAME
        ConfigBoard - configure a board

   SYNOPSIS
        error = ConfigBoard( board, configDev )
        D0                   A0     A1

   FUNCTION
        This routine configures an expansion board.  The board
        will generally live at E_EXPANSIONBASE, but the base is
        passed as a parameter to allow future compatibility.
        The configDev parameter must be a valid configDev that
        has already had ReadExpansionRom() called on it.

        ConfigBoard will allocate expansion memory and place
        the board at its new address.  It will update configDev
        accordingly.  If there is not enough expansion memory
        for this board then an error will be returned.

   INPUTS
        board - the current address that the expansion board is
                responding.
        configDev - an initialized ConfigDev structure, returned
                by AllocConfigDev.

   RESULTS
        error - non-zero if there was a problem configuring this board
                (Can return EE_OK or EE_NOEXPANSION)

   SEE ALSO
        FreeConfigDev
```

```
expansion.library/FindConfigDev                    expansion.library/FindConfigDev

   NAME
        FindConfigDev - find a matching ConfigDev entry

   SYNOPSIS
        configDev = FindConfigDev( oldConfigDev, manufacturer, product )
        D0                         A0            D0            D1

   FUNCTION
        This routine searches the list of existing ConfigDev
        structures in the system and looks for one that has
        the specified manufacturer and product codes.

        If the oldConfigDev is NULL the the search is from the
        start of the list of configuration devices.  If it is
        not null then it searches from the first configuration
        device entry AFTER oldConfigDev.

        A code of -1 is treated as a wildcard -- e.g. it matches
        any manufacturer (or product)

   INPUTS
        oldConfigDev - a valid ConfigDev structure, or NULL to start
                from the start of the list.
        manufacturer - the manufacturer code being searched for, or
                -1 to ignore manufacturer numbers.
        product - the product code being searched for, or -1 to
                ignore product numbers.

   RESULTS
        configDev - the next ConfigDev entry that matches the
                manufacturer and product codes, or NULL if there
                are no more matches.

   EXCEPTIONS

   EXAMPLES
        /* to find all configdevs of the proper type */
        struct ConfigDev *cd = NULL;

        while( cd = FindConfigDev( cd, MANUFACTURER, PRODUCT ) ) {
                /* do something with the returned ConfigDev */
        }

   SEE ALSO

   BUGS
```

```
expansion.library/FreeConfigDev                    expansion.library/FreeConfigDev

   NAME
        FreeConfigDev - free a ConfigDev structure

   SYNOPSIS
        FreeConfigDev( configDev )
                       A0

   FUNCTION
        This routine frees a ConfigDev structure as returned by
        AllocConfigDev.

   INPUTS
        configDev - a valid ConfigDev structure.

   RESULTS

   EXCEPTIONS

   SEE ALSO
        AllocConfigDev

   BUGS
```

```
expansion.library/FreeExpansionMem          expansion.library/FreeExpansionMem

   NAME
        FreeExpansionMem - allocate standard device expansion memory

   SYNOPSIS
        FreeExpansionMem( startSlot, numSlots )
                          D0         D1

   FUNCTION
        (Not typically called by user code)

        This function allocates numslots of expansion space (each slot
        is E_SLOTSIZE bytes).  It is the inverse function of
        AllocExpansionMem().

   INPUTS
        startSlot - the slot number that was allocated, or -1 for error.
        numSlots - the number of slots to be freed.

   RESULTS

   EXAMPLES

   EXCEPTIONS
        If the caller tries to free a slot that is already in the
        free list, FreeExpansionMem will Alert() (e.g. crash the
        system).

   SEE ALSO
        AllocExpansionMem

   BUGS
```

```
expansion.library/GetCurrentBinding        expansion.library/GetCurrentBinding

   NAME
        GetCurrentBinding - sets static board configuration area

   SYNOPSIS
        actual = GetCurrentBinding( currentBinding, size )
                                    A0              D0:16

   FUNCTION
        This function writes the contents of the "currentBinding"
        structure out of a private place.  It may be set via
        SetCurrentBinding().  This is really a kludge, but it is
        the only way to pass extra arguments to a newly configured
        device.

        A CurrentBinding structure has the name of the currently
        loaded file, the product string that was associated with
        this driver, and a pointer to the head of a singly linked
        list of ConfigDev structures (linked through the cd_NextCD
        field).

        Many devices may not need this information; they have hard
        coded into themselves their manufacture number.  It is
        recommended that you at least check that you can deal with
        the product code in the linked ConfigDev structures.

   INPUTS
        currentBinding - a pointer to a CurrentBinding structure

        size - The size of the user's binddriver structure.
            Do not pass in less than sizeof(struct CurrentBinding).

   RESULTS
        actual - the true size of a CurrentBinding structure is returned.

   SEE ALSO
        GetCurrentBinding
```

```
expansion.library/MakeDosNode                    expansion.library/MakeDosNode

   NAME
       MakeDosNode -- construct dos data structures that a disk needs

   SYNOPSIS
       deviceNode = MakeDosNode( parameterPkt )
       D0                        A0

       struct DeviceNode * MakeDosNode( void * );

   FUNCTION
       This routine manufactures the data structures needed to enter
       a dos disk device into the system.  This consists of a DeviceNode,
       a FileSysStartupMsg, a disk environment vector, and up to two
       bcpl strings.  See the libraries/dosextens.h and
       libraries/filehandler.h include files for more information.

       MakeDosNode will allocate all the memory it needs, and then
       link the various structure together.  It will make sure all
       the structures are long-word aligned (as required by the DOS).
       It then returns the information to the user so he can
       change anything else that needs changing.  Typically he will
       then call AddDosNode() to enter the new device into the dos
       tables.

   INPUTS
       parameterPkt - a longword array containing all the information
           needed to initialize the data structures.  Normally I
           would have provided a structure for this, but the variable
           length of the packet caused problems.  The two strings are
           null terminated strings, like all other exec strings.

           longword    description
           --------    -----------
           0           string with dos handler name
           1           string with exec device name
           2           unit number (for OpenDevice)
           3           flags (for OpenDevice)
           4           # of longwords in rest of enviroment
           5-n         file handler environment (see libraries/filehandler.h)

   RESULTS
       deviceNode - pointer to initialize device node structure, or
           null if there was not enough memory.  You may need to change
           certain fields before passing the DeviceNode to AddDosNode().

   EXAMPLES
       /* set up a 3.5" amiga format floppy drive for unit 1 */

       char execName[] = "trackdisk.device";
       char dosName[] = "df1";

       ULONG parmPkt[] = {
           (ULONG) dosName,
           (ULONG) execName,
           1,                  /* unit number */
           0,                  /* OpenDevice flags */

           /* here is the environment block */
           16,                 /* table upper bound */
           512>>2,             /* # longwords in a block */
           0,                  /* sector origin -- unused */
           2,                  /* number of surfaces */
           1,                  /* secs per logical block -- leave as 1 */
           11,                 /* blocks per track */
```

```
           2,                  /* reserved blocks -- 2 boot blocks */
           0,                  /* ?? -- unused */
           0,                  /* interleave */
           0,                  /* lower cylinder */
           79,                 /* upper cylinder */
           5,                  /* number of buffers */
           MEMF_CHIP,          /* type of memory for buffers */
           (~0 >> 1),          /* largest transfer size (largest signed #) */
           ~1,                 /* addmask */
           0,                  /* boot priority */
           0x444f5300,         /* dostype: 'DOS\0' */
       };

       struct Device Node *node, *MakeDosNode();

       node = MakeDosNode( parmPkt );


   SEE ALSO
       AddDosNode, libraries/dosextens.h, libraries/filehandler.h
```

expansion.library

Page 15

```
expansion.library/ObtainConfigBinding   expansion.library/ObtainConfigBinding

    NAME
        ObtainConfigBinding - try to get permission to bind drivers

    SYNOPSIS
        ObtainConfigBinding()


    FUNCTION
        ObtainConfigBinding gives permission to bind drivers to
        ConfigDev structures.  It exists so two drivers at once
        do not try and own the same ConfigDev structure.  This
        call will block until it is safe proceed.

        It is crucially important that people lock out others
        before loading new drivers.  Much of the data that is used
        to configure things is statically kept, and others need
        to be kept from using it.

        This call is built directly on Exec SignalSemaphore code
        (e.g. ObtainSemaphore).

    INPUTS

    RESULTS

    EXCEPTIONS

    SEE ALSO
        ReleaseConfigBinding()

    BUGS
```

expansion.library

Page 16

```
expansion.library/ReadExpansionByte       expansion.library/ReadExpansionByte

    NAME
        ReadExpansionByte - read a byte nybble by nybble.

    SYNOPSIS
        byte = ReadExpansionByte( board, offset )
        D0                        A0     D0

    FUNCTION
        (Not typically called by user code)

        ReadExpansionByte reads a byte from a new-style expansion
        board.  These boards have their readable data organized
        as a series of nybbles in memory.  This routine reads
        two nybbles and returns the byte value.

        In general, this routine will only be called by ReadExpansionRom.

        The offset is a byte offset, as if into a ExpansionRom structure.
        The actual memory address read will be four times larger.
        The macros EROFFSET and ECOFFSET are provided to help get
        these offsets from C.

    INPUTS
        board - a pointer to the base of a new style expansion board.
        offset - a logical offset from the board base

    RESULTS
        byte - a byte of data from the expansion board.

    EXAMPLES
        byte = ReadExpansionByte( cd->BoardAddr, EROFFSET( er_Type ) );
        ints = ReadExpansionByte( cd->BoardAddr, ECOFFSET( ec_Interrupt ) );

    SEE ALSO
        WriteExpansionByte, ReadExpansionRom
```

```
expansion.library/ReadExpansionRom        expansion.library/ReadExpansionRom

   NAME
        ReadExpansionRom - read a boards configuration rom space

   SYNOPSIS
        error = ReadExpansionRom( board, configDev )
        D0                        A0     A1

   FUNCTION
        (Not typically called by user code)

        ReadExpansionRom reads a the rom portion of an expansion
        device in to cd_Rom portion of a ConfigDev structure.
        This routine knows how to detect whether or not there is
        actually a board there,

        In addition, the Rom portion of a new style expansion board
        is encoded in ones-complement format (except for the first
        two nybbles -- the er_Type field).  ReadExpansionRom knows
        about this and un-complements the appropriate fields.

   INPUTS
        board - a pointer to the base of a new style expansion board.
        configDev - the ConfigDev structure that will be read in.
        offset - a logical offset from the configdev base

   RESULTS
        error - If the board address does not contain a valid new style
                expansion board, then error will be non-zero.

   EXAMPLES

        configDev = AllocConfigDev();
        if( ! configDev ) panic();

        error = ReadExpansionBoard( board, configDev );
        if( ! error ) {
                configDev->cd_BoardAddr = board;
                ConfigBoard( configDev );
        }

   SEE ALSO
        ReadExpansionByte, WriteExpansionByte
```

```
expansion.library/ReleaseConfigBinding expansion.library/ReleaseConfigBinding

   NAME
        ReleaseConfigBinding - allow others to bind to drivers

   SYNOPSIS
        ReleaseConfigBinding()


   FUNCTION
        This call should be used when you are done binding drivers
        to ConfigDev entries.  It releases the SignalSemaphore; this
        allows others to bind their drivers to ConfigDev structures.

   SEE ALSO
        ObtainConfigBinding()
```

```
expansion.library/RemConfigDev                 expansion.library/RemConfigDev

   NAME
        RemConfigDev - remove a ConfigDev structure from the system

   SYNOPSIS
        RemConfigDev( configDev )
                      A0

   FUNCTION
        (Not typically called by user code)

        This routine removes the specified ConfigDev structure from the
        list of Configuration Devices in the system.

   INPUTS
        configDev - a valid ConfigDev structure.

   RESULTS

   EXCEPTIONS

   SEE ALSO
        AddConfigDev

   BUGS
```

```
expansion.library/SetCurrentBinding      expansion.library/SetCurrentBinding

   NAME
        SetCurrentBinding - sets static board configuration area

   SYNOPSIS
        SetCurrentBinding( currentBinding, size )
                           A0              D0:16

   FUNCTION
        This function records the contents of the "currentBinding"
        structure in a private place.  It may be read via
        GetCurrentBinding().  This is really a kludge, but it is
        the only way to pass extra arguments to a newly configured
        device.

        A CurrentBinding structure has the name of the currently
        loaded file, the product string that was associated with
        this driver, and a pointer to the head of a singly linked
        list of ConfigDev structures (linked through the cd_NextCD
        field).

        Many devices may not need this information; they have hard
        coded into themselves their manufacture number.  It is
        recommended that you at least check that you can deal with
        the product code in the linked ConfigDev structures.

   INPUTS
        currentBinding - a pointer to a CurrentBinding structure

        size - The size of the user's binddriver structure.  No
            more than this much data will be copied.  If size is
            less than the library's idea a CurrentBinding size,
            then the library's structure will be null padded.

   SEE ALSO
        GetCurrentBinding
```

```
expansion.library/WriteExpansionByte     expansion.library/WriteExpansionByte

   NAME
        WriteExpansionByte - write a byte nybble by nybble.

   SYNOPSIS
        WriteExpansionByte( board, offset, byte )
                            A0     D0      D1

   FUNCTION
        (Not typically called by user code)

        WriteExpansionByte writes a byte to a new-style expansion
        board.  These boards have their writeable data organized
        as a series of nybbles in memory.  This routine writes
        two nybbles in a very careful manner to work with all
        types of new expansion boards.

        To make certain types of board less expensive, an expansion
        board's write registers may be organized as either a
        byte-wide or nybble-wide register.  If it is nybble-wide
        then it must latch the less significant nybble until the
        more significant nybble is written.  This allows the
        following algorithm to work with either type of board:

                write the low order nybble to bits D15-D12 of
                        byte (offset*4)+2

                write the entire byte to bits D15-D8 of
                        byte (offset*4)

        The offset is a byte offset into a ExpansionRom structure.
        The actual memory address read will be four times larger.
        The macros EROFFSET and ECOFFSET are provided to help get
        these offsets from C.

   INPUTS
        board - a pointer to the base of a new style expansion board.
        offset - a logical offset from the configdev base
        byte - the byte of data to be written to the expansion board.

   EXAMPLES
        WriteExpansionByte( cd->BoardAddr, ECOFFSET( ec_Shutup ),  0 );
        WriteExpansionByte( cd->BoardAddr, ECOFFSET( ec_Interrupt ), 1 );

   SEE ALSO
        ReadExpansionByte, ReadExpansionRom
```

TABLE OF CONTENTS

gadtools.library/CreateContext
gadtools.library/CreateGadgetA
gadtools.library/CreateMenusA
gadtools.library/DrawBevelBoxA
gadtools.library/FreeGadgets
gadtools.library/FreeMenus
gadtools.library/FreeVisualInfo
gadtools.library/GetVisualInfoA
gadtools.library/GT_BeginRefresh
gadtools.library/GT_EndRefresh
gadtools.library/GT_FilterIMsg
gadtools.library/GT_GetIMsg
gadtools.library/GT_PostFilterIMsg
gadtools.library/GT_RefreshWindow
gadtools.library/GT_ReplyIMsg
gadtools.library/GT_SetGadgetAttrsA
gadtools.library/LayoutMenuItemsA
gadtools.library/LayoutMenusA

```
gadtools.library/CreateContext                   gadtools.library/CreateContext

   NAME
        CreateContext -- Create a place for GadTools context data. (V36)

   SYNOPSIS
        gad = CreateContext(glistpointer);
        D0                  A0

        struct Gadget *CreateContext(struct Gadget **);

   FUNCTION
        Creates a place for GadTools to store any context data it might
        need for your window.  In reality, an unselectable invisible
        gadget is created, with room for the context data.
        This function also establishes the linkage from a glist type
        pointer to the individual gadget pointers.  Call this function
        before any of the other gadget creation calls.

   INPUTS
        glistptr - Address of a pointer to a Gadget, which was previously
            set to NULL.  When all the gadget creation is done, you may
            use that pointer as your NewWindow.FirstGadget, or
            in intuition.library/AddGList(), intuition.library/RefreshGList(),
            FreeGadgets(), etc.

   RESULT
        gad - Pointer to context gadget, or NULL if failure.

   EXAMPLE

        struct Gadget *gad;
        struct Gadget *glist = NULL;
        gad = CreateContext(&glist);
        /*  Other creation calls go here */
        if (gad)
        {
            myNewWindow.FirstGadget = glist;
            if ( myWindow = OpenWindow(&myNewWindow) )
            {
                GT_RefreshWindow(win);
                /* other stuff */
                CloseWindow(myWindow);
            }
        }
        FreeGadgets(glist);

   NOTES

   BUGS

   SEE ALSO
```

gadtools.library Page 3

```
gadtools.library/CreateGadgetA                  gadtools.library/CreateGadgetA

   NAME
       CreateGadgetA -- Allocate and initialize a gadtools gadget. (V36)
       CreateGadget -- Varargs stub for CreateGadgetA(). (V36)

   SYNOPSIS
       gad = CreateGadgetA(kind, previous, newgad, taglist)
       D0                  D0    A0        A1      A2

       struct Gadget *CreateGadgetA(ULONG, struct Gadget *,
           struct NewGadget *, struct TagItem *);

       gad = CreateGadget(kind, previous, newgad, firsttag, ...)

       struct Gadget *CreateGadget(ULONG, struct Gadget *,
           struct NewGadget *, Tag, ...);

   FUNCTION
       CreateGadgetA() allocates and initializes a new gadget of the
       specified kind, and attaches it to the previous gadget.  The
       gadget is created based on the supplied kind, NewGadget structure,
       and tags.

   INPUTS
       kind - to indicate what kind of gadget is to be created.
       previous - pointer to the previous gadget that this new gadget
           is to be attached to.
       newgad - a filled in NewGadget structure describing the desired
           gadget's size, position, label, etc.
       taglist - pointer to a TagItem list.

   TAGS
       All kinds:
       GT_Underscore (GadTools V37 and higher only).
           Indicates the symbol that precedes the character in the gadget
           label to be underscored.  This would be to indicate keyboard
           equivalents for gadgets (note that GadTools does not process
           the keys - it just displays the underscore).
           Example: To underscore the "M" in "Mode"...
               ng.ng_GadgetText = "_Mode:";
               gad = CreateGadget(..._KIND, &ng, prev,
                       GT_Underscore, '_',
                       ...
                       );

       BUTTON_KIND (action buttons):
       GA_Disabled (BOOL) - Set to TRUE to disable gadget, FALSE otherwise
           (defaults to FALSE).

       CHECKBOX_KIND (on/off items):
       GTCB_Checked (BOOL) - Initial state of checkbox, defaults to FALSE.
       GA_Disabled (BOOL) - Set to TRUE to disable gadget, FALSE otherwise
           (defaults to FALSE).

       CYCLE_KIND (multiple state selections):
       GTCY_Labels (STRPTR *) - Pointer to NULL-terminated array of strings
           that are the choices offered by the cycle gadget (required).
       GTCY_Active (UWORD) - The ordinal number (counting from zero) of
           the initially active choice of a cycle gadget (defaults to zero).
       GA_Disabled (BOOL) - (GadTools V37 and higher only)
           Set to TRUE to disable gadget, FALSE otherwise
           (defaults to FALSE).

       INTEGER_KIND (numeric entry):
       GTIN_Number (ULONG) - The initial contents of the integer gadget
```

gadtools.library Page 4

```
           (default zero).
       GTIN_MaxChars (UWORD) - The maximum number of digits that the
           integer gadget is to hold (defaults to 10).
       GA_Disabled (BOOL) - Set to TRUE to disable gadget, FALSE otherwise
           (defaults to FALSE).
       STRINGA_ExitHelp (BOOL) - (New for V37) Set to TRUE to have the
           help-key cause an exit from the integer gadget.  You will
           then receive a GADGETUP with code = 0x5F (rawkey for help).
       GA_TabCycle (BOOL) - (New for V37) Set to TRUE so that pressing
           <TAB> or <Shift-TAB> will activate the next or previous
           such gadget. (defaults to TRUE, unlike regular Intuition string
           gadgets, which default to FALSE).

       LISTVIEW_KIND (scrolling list):
       GTLV_Top (UWORD) - Top item visible in the listview (defaults to zero).
       GTLV_Labels (struct List *) - List of labels whose ln_Name fields
           are to be displayed in the listview.
       GTLV_ReadOnly (BOOL) - If TRUE, then listview is read-only.
       GTLV_ScrollWidth (UWORD) - Width of scroll bar for listview.
           Must be greater than zero (defaults to 16).
       GTLV_ShowSelected (struct Gadget *) - NULL to have the currently
           selected item displayed beneath the listview, or pointer to
           an already-created GadTools STRING_KIND gadget to have an
           editable display of the currently selected item.
       GTLV_Selected (UWORD) - Ordinal number of currently selected
           item, or ~0 to have no current selection (defaults to ~0).
       LAYOUTA_Spacing - Extra space to place between lines of listview
           (defaults to zero).

       MX_KIND (mutually exclusive, radio buttons):
       GTMX_Labels (STRPTR *) - Pointer to a NULL-terminated array of
           strings which are to be the labels beside each choice in a
           set of mutually exclusive gadgets.
       GTMX_Active (UWORD) - The ordinal number (counting from zero) of
           the initially active choice of an mx gadget (Defaults to zero).
       GTMX_Spacing (UWORD) - The amount of space between each choice
           of a set of mutually exclusive gadgets.  This amount is added
           to the font height to produce the vertical shift between
           choices. (defaults to one).
       LAYOUTA_Spacing - FOR COMPATIBILITY ONLY.  Use GTMX_Spacing instead.
           The number of extra pixels to insert between
           each choice of a mutually exclusive gadget.  This is added
           to the present gadget image height (9) to produce the
           true spacing between choices.  (defaults to
           FontHeight-8, which is zero for 8-point font users).

       NUMBER_KIND (read-only numeric):
       GTNM_Number - A signed long integer to be displayed as a read-only
           number (default 0).
       GTNM_Border (BOOL) - If TRUE, this flag asks for a recessed
           border to be placed around the gadget.

       PALETTE_KIND (color selection):
       GTPA_Depth (UWORD) - Number of bitplanes in the palette
           (defaults to 1).
       GTPA_Color (UBYTE) - Initially selected color of the palette
           (defaults to 1).
       GTPA_ColorOffset (UBYTE) - First color to use in palette
           (defaults to zero).
       GTPA_IndicatorWidth (UWORD) - The desired width of the current-color
           indicator, if you want one to the left of the palette.
       GTPA_IndicatorHeight (UWORD) - The desired height of the current-color
           indicator, if you want one above the palette.
       GA_Disabled (BOOL) - Set to TRUE to disable gadget, FALSE otherwise
           (defaults to FALSE).
```

```
SCROLLER_KIND (for scrolling through areas or lists):
GTSC_Top (WORD) - Top visible in area scroller represents
    (defaults to zero).
GTSC_Total (WORD) - Total in area scroller represents
    (defaults to zero).
GTSC_Visible (WORD) - Number visible in scroller (defaults to 2).
GTSC_Arrows (UWORD) - Asks for arrows to be attached to the scroller.
    The value supplied will be taken as the width of each arrow button
    for a horizontal scroller, or the height of each button for a
    vertical scroller (the other dimension will match the whole
    scroller).
PGA_Freedom - Whether scroller is horizontal or vertical.
    Choose LORIENT_VERT or LORIENT_HORIZ (defaults to horiz).
GA_Immediate (BOOL) - Hear every IDCMP_GADGETDOWN event from scroller
    (defaults to FALSE).
GA_RelVerify (BOOL) - Hear every IDCMP_GADGETUP event from scroller
    (defaults to FALSE).
GA_Disabled (BOOL) - Set to TRUE to disable gadget, FALSE otherwise
    (defaults to FALSE).

SLIDER_KIND (to indicate level or intensity):
GTSL_Min (WORD) - Minimum level for slider (default 0).
GTSL_Max (WORD) - Maximum level for slider (default 15).
GTSL_Level (WORD) - Current level of slider (default 0).
GTSL_MaxLevelLen (UWORD) - Max. length in characters of level string
    when rendered beside slider.
GTSL_LevelFormat (STRPTR) - C-Style formatting string for slider
    level.  Be sure to use the 'l' (long) modifier.  This string
    is processed using exec/RawDoFmt(), so refer to that function
    for details.
GTSL_LevelPlace - One of PLACETEXT_LEFT, PLACETEXT_RIGHT,
    PLACETEXT_ABOVE, or PLACETEXT_BELOW, indicating where the level
    indicator is to go relative to slider (default to PLACETEXT_LEFT).
GTSL_DispFunc ( LONG (*function)(struct Gadget *, WORD) ) - Function
    to calculate level to be displayed.  A number-of-colors slider
    might want to set the slider up to think depth, and have a
    (1 << n) function here.  Defaults to none.  Your function must
    take a pointer to gadget as the first parameter, the level
    (a WORD) as the second, and return the result as a LONG.
GA_Immediate (BOOL) - If you want to hear each slider IDCMP_GADGETDOWN
    event.
GA_RelVerify (BOOL) - If you want to hear each slider IDCMP_GADGETUP
    event.
PGA_Freedom - Set to LORIENT_VERT or LORIENT_HORIZ to have a
    vertical or horizontal slider.
GA_Disabled (BOOL) - Set to TRUE to disable gadget, FALSE otherwise
    (defaults to FALSE).

STRING_KIND (text-entry):
GTST_String (STRPTR) - The initial contents of the string gadget,
    or NULL (default) if string is to start empty.
GTST_MaxChars (UWORD) - The maximum number of characters that the
    string gadget is to hold.
GA_Disabled (BOOL) - Set to TRUE to disable gadget, FALSE otherwise
    (defaults to FALSE).
STRINGA_ExitHelp (BOOL) - (New for V37) Set to TRUE to have the
    help-key cause an exit from the string gadget.  You will
    then receive a GADGETUP with code = 0x5F (rawkey for help).
GA_TabCycle (BOOL) - (New for V37) Set to TRUE so that pressing
    <TAB> or <Shift-TAB> will activate the next or previous
    such gadget. (defaults to TRUE, unlike regular Intuition string
    gadgets, which default to FALSE).

TEXT_KIND (read-only text):
GTTX_Text - Pointer to a NULL terminated string to be displayed,
    as a read-only text-display gadget, or NULL. defaults to NULL.
```

```
    GTTX_CopyText (BOOL) -  This flag instructs the text-display gadget
        to copy the supplied text string, instead of using only
        pointer to the string.  This only works for the initial value
        of GTTX_Text set at CreateGadget() time.  If you subsequently
        change GTTX_Text, the new text will be referenced by pointer,
        not copied.  Do not use this tag with a NULL GTTX_Text.
    GTTX_Border (BOOL) - If TRUE, this flag asks for a recessed
        border to be placed around the gadget.


RESULT
    gad - pointer to the new gadget, or NULL if the allocation failed
        or if previous was NULL.

EXAMPLE

NOTES
    Note that the ng_VisualInfo and ng_TextAttr fields of the
    NewGadget structure must be set to valid VisualInfo and
    TextAttr pointers, or this function will fail.

    Starting with V37, string and integer gadgets have the GFLG_TABCYCLE
    feature automatically.  If the user presses Tab or Shift-Tab while
    in a string or integer gadget, the next or previous one in
    sequence will be activated.  You will hear a GADGETUP with a code
    of 0x09.  Use {GA_TabCycle, FALSE} to supress this.


BUGS

SEE ALSO
    FreeGadgets(), GT_SetGadgetAttrs(), GetVisualInfo().
```

```
gadtools.library/CreateMenusA                       gadtools.library/CreateMenusA

   NAME
        CreateMenusA -- Allocate and fill out a menu structure. (V36)
        CreateMenus -- Varargs stub for CreateMenus(). (V36)

   SYNOPSIS
        menu = CreateMenusA(newmenu, taglist)
        D0                  A0       A1

        struct Menu *CreateMenusA(struct NewMenu *, struct TagItem *);

        menu = CreateMenus(newmenu, firsttag, ...)

        struct Menu *CreateMenus(struct NewMenu *, Tag, ...);

   FUNCTION
        CreateMenusA() allocates and initializes a complete menu
        structure based on the supplied array of NewMenu structures.
        Optionally, CreateMenusA() can allocate and initialize a complete
        set of menu items and sub-items for a single menu title.  This
        is dictated by the contents of the array of NewMenus.

   INPUTS
        newmenu - Pointer to an array of initialized struct NewMenus.
        taglist - Pointer to a TagItem list.

   TAGS
        GTMN_FrontPen (UBYTE) - Pen number to be used for menu text.
            (defaults to zero).
        GTMN_FullMenu (BOOL) - (GadTools V37 and higher only)
            Requires that the NewMenu specification describes a complete
            menu strip, not a fragment.  If a fragment is found,
            CreateMenusA() will fail with a secondary error of
            GTMENU_INVALID.  (defaults to FALSE).
        GTMN_SecondaryError (ULONG *) - (GadTools V37 and higher only)
            Supply a pointer to a NULL-initialized ULONG to receive a
            descriptive error code.  Possible values:
            GTMENU_INVALID - NewMenu structure describes an illegal
                menu.  (CreateMenusA() will fail with a NULL result).
            GTMENU_TRIMMED - NewMenu structure has too many menus, items,
                or subitems (CreateMenusA() will succeed, returning a
                trimmed-down menu structure).
            GTMENU_NOMEM - CreateMenusA() ran out of memory.

   RESULT
        menu - Pointer to the resulting initialized menu structure (or
            the resulting FirstItem), with all the links for menu items
            and subitems in place.
            The result will be NULL if CreateMenusA() could not allocate
            memory for the menus, or if the NewMenu array had an
            illegal arrangement (eg. NM_SUB following NM_TITLE).
            (see also the GTMN_SecondaryError tag above).

   EXAMPLE

   NOTES
        The strings you supply for menu text are not copied, and must
        be preserved for the life of the menu.
        The resulting menus have no positional information.  You will
        want to call LayoutMenusA() (or LayoutMenuItemsA()) to supply that.
        CreateMenusA() automatically provides you with a UserData field
        for each menu, menu-item or sub-item.  Use the GTMENU_USERDATA(menu)
        or GTMENUITEM_USERDATA(menuitem) macro to access it.

   BUGS
```

```
   SEE ALSO
        LayoutMenusA(), FreeMenus(), gadtools.h/GTMENU_USERDATA(),
        gadtools.h/GTMENUITEM_USERDATA()
```

```
gadtools.library/DrawBevelBoxA                  gadtools.library/DrawBevelBoxA

   NAME
       DrawBevelBoxA -- Draws a bevelled box. (V36)
       DrawBevelBox -- Varargs stub for DrawBevelBox(). (V36)

   SYNOPSIS
       DrawBevelBoxA(rport, left, top, width, height, taglist)
                     A0     D0    D1   D2     D3      A3

       VOID DrawBevelBoxA(struct RastPort *, WORD, WORD, WORD, WORD,
           struct TagItem *taglist);

       DrawBevelBox(rport, left, top, width, height, firsttag, ...)

       VOID DrawBevelBox(struct RastPort *, WORD, WORD, WORD, WORD,
           Tag, ...);

   FUNCTION
       DrawBevelBoxA() renders a bevelled box of specified dimensions
       into the supplied RastPort.

   INPUTS
       rport - The RastPort into which the box is to be drawn.
       left - The left edge of the box.
       top - The top edge of the box.
       width - The width of the box.
       height - The height of the box.
       taglist - Pointer to a TagItem list.

   TAGS
       GTBB_Recessed (BOOL):  Set to anything for a recessed-looking box.
           If absent, the box defaults, it would be raised.
       GT_VisualInfo (APTR):  You MUST supply the value you obtained
           from an earlier call to GetVisualInfoA().

   RESULT
       None.

   EXAMPLE

   NOTES
       DrawBevelBox() is a rendering operation, not a gadget.  That
       means you must refresh it at the appropriate time, like any
       other rendering operation.

   BUGS

   SEE ALSO
       GetVisualInfoA()
```

```
gadtools.library/FreeGadgets                      gadtools.library/FreeGadgets

   NAME
       FreeGadgets -- Free a linked list of gadgets. (V36)

   SYNOPSIS
       FreeGadgets(glist)
                   A0

       VOID FreeGadgets(struct Gadget *glist);
                        A0

   FUNCTION
       Frees any GadTools gadgets found on the linked list of gadgets
       beginning with the specified one.  Frees all the memory that was
       allocated by CreateGadgetA().  This function will return safely
       with no action if it receives a NULL parameter.

   INPUTS
       glist - pointer to first gadget in list to be freed.

   RESULT
       none

   EXAMPLE

   NOTES

   BUGS

   SEE ALSO
       CreateGadgetA()
```

gadtools.library Page 11

```
gadtools.library/FreeMenus                             gadtools.library/FreeMenus

   NAME
       FreeMenus -- Frees memory allocated by CreateMenusA(). (V36)

   SYNOPSIS
       FreeMenus(menu)
                 A0

       VOID FreeMenus(struct Menu *);

   FUNCTION
       Frees the menus allocated by CreateMenusA().  It is safe to
       call this function with a NULL parameter.

   INPUTS
       menu - Pointer to menu structure (or first MenuItem) obtained
       from CreateMenusA().

   RESULT
       None.

   EXAMPLE

   NOTES

   BUGS

   SEE ALSO
       CreateMenusA()
```

gadtools.library Page 12

```
gadtools.library/FreeVisualInfo                   gadtools.library/FreeVisualInfo

   NAME
       FreeVisualInfo -- Return any resources taken by GetVisualInfo. (V36)

   SYNOPSIS
       FreeVisualInfo(vi)
                      A0

       VOID FreeVisualInfo(APTR);

   FUNCTION
       FreeVisualInfo() returns any memory or other resources that
       were allocated by GetVisualInfoA().  You should only call this function
       once you are done with using the gadgets (i.e. after CloseWindow()),
       but while the screen is still valid (i.e. before CloseScreen() or
       UnlockPubScreen()).

   INPUTS
       vi - Pointer that was obtained by calling GetVisualInfoA().

   RESULT
       None.

   EXAMPLE

   NOTES

   BUGS

   SEE ALSO
       GetVisualInfoA()
```

```
gadtools.library/GetVisualInfoA                    gadtools.library/GetVisualInfoA

   NAME
        GetVisualInfoA -- Get information GadTools needs for visuals. (V36)
        GetVisualInfo -- Varargs stub for GetVisualInfoA(). (V36)

   SYNOPSIS
        vi = GetVisualInfoA(screen, taglist)
        D0                  A0      A1

        APTR vi = GetVisualInfoA(struct Screen *, struct TagItem *);

        vi = GetVisualInfo(screen, firsttag, ...)

        APTR vi = GetVisualInfo(struct Screen *, Tag, ...);

   FUNCTION
        Get a pointer to a (private) block of data containing various bits
        of information that GadTools needs to ensure the best quality
        visuals.  Use the result in the NewGadget structure of any gadget
        you create, or as a parameter to the various menu calls.  Once the
        gadgets/menus are no longer needed (after the last CloseWindow),
        call FreeVisualInfo().

   INPUTS
        screen - Pointer to the screen you will be opening on.
        taglist - Pointer to list of TagItems.

   RESULT
        vi - Pointer to private data.

   EXAMPLE

   NOTES

   BUGS

   SEE ALSO
        FreeVisualInfo(), intuition/LockPubScreen(),
        intuition/UnlockPubScreen()
```

```
gadtools.library/GT_BeginRefresh                  gadtools.library/GT_BeginRefresh

   NAME
        GT_BeginRefresh -- Begin refreshing friendly to GadTools. (V36)

   SYNOPSIS
        GT_BeginRefresh(win)
                        A0

        VOID GT_BeginRefresh(struct Window *);

   FUNCTION
        Invokes the intuition.library/BeginRefresh() function in a manner
        friendly to the Gadget Toolkit.  This function call permits the
        GadTools gadgets to refresh themselves at the correct time.
        Call GT_EndRefresh() function when done.

   INPUTS
        win - Pointer to Window structure for which a IDCMP_REFRESHWINDOW
            IDCMP event was received.

   RESULT
        None.

   EXAMPLE

   NOTES
        The nature of GadTools precludes the use of the IDCMP flag
        WFLG_NOCAREREFRESH.  You must handle IDCMP_REFRESHWINDOW events
        in at least the minimal way, namely:

                case IDCMP_REFRESHWINDOW:
                    GT_BeginRefresh(win);
                    GT_EndRefresh(win, TRUE);
                    break;

   BUGS

   SEE ALSO
        intuition.library/BeginRefresh()
```

gadtools.library Page 15

```
gadtools.library/GT_EndRefresh                 gadtools.library/GT_EndRefresh

   NAME
       GT_EndRefresh -- End refreshing friendly to GadTools. (V36)

   SYNOPSIS
       GT_EndRefresh(win, complete)
                     A0   D0

       VOID GT_EndRefresh(struct Window *, BOOL complete);

   FUNCTION
       Invokes the intuition.library/EndRefresh() function in a manner
       friendly to the Gadget Toolkit.  This function call permits
       GadTools gadgets to refresh themselves at the correct time.
       Call this function to EndRefresh() when you have used
       GT_BeginRefresh().

   INPUTS
       win - Pointer to Window structure for which a IDCMP_REFRESHWINDOW
           IDCMP event was received.
       complete - TRUE when done with refreshing.

   RESULT
       None.

   EXAMPLE

   NOTES

   BUGS

   SEE ALSO
       intuition.library/EndRefresh()
```

gadtools.library Page 16

```
gadtools.library/GT_FilterIMsg                 gadtools.library/GT_FilterIMsg

   NAME
       GT_FilterIMsg -- Filter an IntuiMessage through GadTools. (V36)

   SYNOPSIS
       modimsg = GT_FilterIMsg(imsg)
       D0                      A1

       struct IntuiMessage *GT_FilterIMsg(struct IntuiMessage *);

   FUNCTION
       NOTE WELL:  Extremely few programs will actually need this function.
       You almost certainly should be using GT_GetIMsg() and GT_ReplyIMsg()
       only, and not GT_FilterIMsg() and GT_PostFilterIMsg().

       GT_FilterIMsg() takes the supplied IntuiMessage and asks the
       Gadget Toolkit to consider and possibly act on it.  Returns
       NULL if the message was only of significance to a GadTools gadget
       (i.e. not to you), else returns a pointer to a modified IDCMP
       message, which may contain additional information.
       You should examine the Class, Code, and IAddress fields of
       the returned message to learn what happened.  Do not make
       interpretations based on the original imsg.
       You should use GT_PostFilterIMsg() to revert to the original
       IntuiMessage once you are done with the modified one.

   INPUTS
       imsg - An IntuiMessage you obtained from a Window's UserPort.

   RESULT
       modimsg - A modified IntuiMessage, possibly with extra information
           from GadTools, or NULL.

   EXAMPLE

   NOTES

   BUGS

   SEE ALSO
       GT_GetIMsg(), GT_PostFilterIMsg()
```

```
gadtools.library/GT_GetIMsg                           gadtools.library/GT_GetIMsg

   NAME
        GT_GetIMsg -- Get an IntuiMessage, with GadTools processing. (V36)

   SYNOPSIS
        imsg = GT_GetIMsg(intuiport)
        D0                A0

        struct IntuiMessage *GT_GetIMsg(struct MsgPort *);

   FUNCTION
        Use GT_GetIMsg() in place of the usual exec.library/GetMsg() when
        reading IntuiMessages from your window's UserPort.  If needed,
        the GadTools dispatcher will be invoked, and suitable processing
        will be done for gadget actions.  This function returns a pointer
        to a modified IntuiMessage (which is a copy of the original,
        possibly with some supplementary information from GadTools).
        If there are no messages (or if the only messages are meaningful
        only to GadTools, NULL will be returned.

   INPUTS
        intuiport - The Window->UserPort of a window that is using the
            Gadget Toolkit.

   RESULT
        imsg - Pointer to modified IntuiMessage, or NULL if there are
            no applicable messages.

   EXAMPLE

   NOTES
        Be sure to use GT_ReplyIMsg() and not exec.library/ReplyMsg() on
        messages obtained with GT_GetIMsg().
        If you intend to do more with the resulting message than read
        its fields, act on it, and reply it, you may find GT_FilterIMsg()
        more appropriate.

   BUGS

   SEE ALSO
        GT_ReplyIMsg(), GT_FilterIMsg()
```

```
gadtools.library/GT_PostFilterIMsg             gadtools.library/GT_PostFilterIMsg

   NAME
        GT_PostFilterIMsg -- Return the unfiltered message after
            GT_FilterIMsg() was called, and clean up. (V36)

   SYNOPSIS
        imsg = GT_PostFilterIMsg(modimsg)
        D0                       A1

        struct IntuiMessage *GT_PostFilterIMsg(struct IntuiMessage *);

   FUNCTION
        NOTE WELL:  Extremely few programs will actually need this function.
        You almost certainly should be using GT_GetIMsg() and GT_ReplyIMsg()
        only, and not GT_FilterIMsg() and GT_PostFilterIMsg().

        Performs any clean-up necessitated by a previous call to
        GT_FilterIMsg().  The original IntuiMessage is now yours to handle.
        Do not interpret the fields of the original IntuiMessage, but
        rather use only the one you got from GT_FilterIMsg().  You
        may only do message related things at this point, such as queueing
        it up or replying it.  Since you got the message with
        exec.library/GetMsg(), your responsibilities do include replying
        it with exec.library/ReplyMsg(). This function may be safely
        called with a NULL parameter.

   INPUTS
        modimsg - A modified IntuiMessage obtained with GT_FilterIMsg().

   RESULT
        imsg - A pointer to the original IntuiMessage, if GT_FilterIMsg()
            returned non-NULL.

   EXAMPLE

   NOTES
        Be sure to use exec.library/ReplyMsg() on the original IntuiMessage
        you obtained with GetMsg(), (which is the what you passed to
        GT_FilterIMsg()), and not on the parameter of this function.

   BUGS

   SEE ALSO
        GT_FilterIMsg()
```

gadtools.library Page 19

```
gadtools.library/GT_RefreshWindow          gadtools.library/GT_RefreshWindow

   NAME
       GT_RefreshWindow -- Refresh all the GadTools gadgets. (V36)

   SYNOPSIS
       GT_RefreshWindow(win, req)
                        A0   A1

       VOID GT_RefreshWindow(struct Window *, struct Requester *);

   FUNCTION
       Perform the initial refresh of all the GadTools gadgets you have
       created.  After you have opened your window, you must call this
       function.  Or, if you have opened your window without gadgets,
       you add the gadgets with intuition.library/AddGList(),
       refresh them using intuition.library/RefreshGList(), then call
       this function.
       You should not need this function at other times.

   INPUTS
       win - Pointer to the Window containing GadTools gadgets.
       req - Pointer to requester, or NULL if not a requester (currently
           ignored - use NULL).

   RESULT
       None.

   EXAMPLE

   NOTES
       req must currently be NULL.  GadTools gadgets are not supported
       in requesters.  This field may allow such support at a future date.

   BUGS

   SEE ALSO
       GT_BeginRefresh()
```

gadtools.library Page 20

```
gadtools.library/GT_ReplyIMsg                      gadtools.library/GT_ReplyIMsg

   NAME
       GT_ReplyIMsg -- Reply a message obtained with GT_GetIMsg(). (V36)

   SYNOPSIS
       GT_ReplyIMsg(imsg)
                    A1

       VOID GT_ReplyIMsg(struct IntuiMessage *);

   FUNCTION
       Reply a modified IntuiMessage obtained with GT_GetIMsg().
       If you use GT_GetIMsg(), use this function where you would normally
       have used exec.library/ReplyMsg().
       You may safely call this routine with a NULL pointer (nothing
       will be done).

   INPUTS
       imsg - A modified IntuiMessage obtained with GT_GetIMsg().

   RESULT
       None.

   EXAMPLE

   NOTES
       When using GadTools, you MUST explicitly GT_ReplyIMsg()
       all messages you receive.  You cannot depend on CloseWindow()
       to handle messages you have not replied.

   BUGS

   SEE ALSO
       GT_GetIMsg()
```

```
gadtools.library/GT_SetGadgetAttrsA       gadtools.library/GT_SetGadgetAttrsA

   NAME
        GT_SetGadgetAttrsA -- Change the attributes of a GadTools gadget. (V36)
        GT_SetGadgetAttrs -- Varargs stub for GT_SetGadgetAttrsA(). (V36)

   SYNOPSIS
        GT_SetGadgetAttrsA(gad, win, req, taglist)
                           A0   A1   A2   A3

        VOID GT_SetGadgetAttrsA(struct Gadget *, struct Window *,
            struct Requester *, struct TagItem *);

        GT_SetGadgetAttrs(gad, win, req, firsttag, ...)

        VOID GT_SetGadgetAttrs(struct Gadget *, struct Window *,
            struct Requester *, Tag, ...);

   FUNCTION
        Change the attributes of the specified gadget, according to the
        attributes chosen in the tag list.

   INPUTS
        gad - Pointer to the gadget in question.
        win - Pointer to the window containing the gadget.
        req - Pointer to the requester containing the gadget, or NULL if
            not in a requester.  (Not yet implemented, use NULL).
        taglist - Pointer to TagItem list.

   TAGS
        BUTTON_KIND:
        GA_Disabled (BOOL) - Set to TRUE to disable gadget, FALSE otherwise
            (defaults to FALSE).

        CHECKBOX_KIND:
        GTCB_Checked (BOOL) - Initial state of checkbox, defaults to FALSE.
        GA_Disabled (BOOL) - Set to TRUE to disable gadget, FALSE otherwise
            (defaults to FALSE).

        CYCLE_KIND:
        GTCY_Active (UWORD) - The ordinal number (counting from zero) of
            the active choice of a cycle gadget (defaults to zero).
        GTCY_Labels (STRPTR *) - (GadTools V37 and higher only)
            Pointer to NULL-terminated array of strings
            that are the choices offered by the cycle gadget.
        GA_Disabled (BOOL) - (GadTools V37 and higher only)
            Set to TRUE to disable gadget, FALSE otherwise
            (defaults to FALSE).

        INTEGER_KIND:
        GTIN_Number (ULONG) - The initial contents of the integer gadget
            (default zero).
        GA_Disabled (BOOL) - Set to TRUE to disable gadget, FALSE otherwise
            (defaults to FALSE).

        LISTVIEW_KIND:
        GTLV_Top (UWORD) - Top item visible in the listview (defaults to zero).
        GTLV_Labels (struct List *) - List of labels whose ln_Name fields
            are to be displayed in the listview.  Use a value of ~0 to
            "detach" your List from the display.  You must detach your list
            before modifying the List structure, since GadTools reserves
            the right to traverse it on another task's schedule.  When you
            are done, attach the list by using the tag pair
            {GTLV_Labels, list}.
        GTLV_Selected (UWORD) - Ordinal number of currently selected
            item (defaults to zero if GTLV_ShowSelected is set).
```

```
        MX_KIND:
        GTMX_Active (UWORD) - The ordinal number (counting from zero) of
            the active choice of an mx gadget (Defaults to zero).

        NUMBER_KIND:
        GTNM_Number - A signed long integer to be displayed as a read-only
            number (default 0).

        PALETTE_KIND:
        GTPA_Color (UBYTE) - Initially selected color of the palette
            (defaults to 1).
        GA_Disabled (BOOL) - Set to TRUE to disable gadget, FALSE otherwise
            (defaults to FALSE).

        SCROLLER_KIND:
        GTSC_Top (WORD) - Top visible in scroller (defaults to zero).
        GTSC_Total (WORD) - Total in scroller area (defaults to zero).
        GTSC_Visible (WORD) - Number visible in scroller (defaults to 2).
        GA_Disabled (BOOL) - Set to TRUE to disable gadget, FALSE otherwise
            (defaults to FALSE).

        SLIDER_KIND:
        GTSL_Min (WORD) - Minimum level for slider (default 0).
        GTSL_Max (WORD) - Maximum level for slider (default 15).
        GTSL_Level (WORD) - Current level of slider (default 0).
        GA_Disabled (BOOL) - Set to TRUE to disable gadget, FALSE otherwise
            (defaults to FALSE).

        STRING_KIND:
        GTST_String (STRPTR) - The initial contents of the string gadget,
            or NULL (default) if string is to start empty.
        GA_Disabled (BOOL) - Set to TRUE to disable gadget, FALSE otherwise
            (defaults to FALSE).

        TEXT_KIND:
        GTTX_Text - Pointer to a NULL terminated string to be displayed,
            as a read-only text-display gadget, or NULL. defaults to NULL.

   RESULT
        None.

   EXAMPLE

   NOTES
        req must currently be NULL.  GadTools gadgets are not supported
        in requesters.  This field may allow such support at a future date.

        This function may not be called inside of a GT_BeginRefresh() /
        GT_EndRefresh() session.  (As always, restrict yourself to simple
        rendering functions).

   BUGS

   SEE ALSO
```

gadtools.library

```
gadtools.library/LayoutMenuItemsA          gadtools.library/LayoutMenuItemsA

   NAME
       LayoutMenuItemsA -- Position all the menu items. (V36)
       LayoutMenuItems -- Varargs stub for LayoutMenuItemsA(). (V36)

   SYNOPSIS
       success = LayoutMenuItemsA(menuitem, vi, taglist)
       D0                         A0        A1  A2

       BOOL LayoutMenuItemsA(struct MenuItem *, APTR, struct TagItem *);

       success = LayoutMenuItems(menuitem, vi, firsttag, ...)

       BOOL LayoutMenuItemsA(struct MenuItem *, APTR, Tag, ...);

   FUNCTION
       Lays out all the menu items and sub-items according to
       the supplied visual information and tag parameters.  You would use this
       if you used CreateMenusA() to make a single menu-pane (with sub-items,
       if any), instead of a whole menu strip.
       This routine attempts to columnize and/or shift the MenuItems in
       the event that a menu would be too tall or too wide.

   INPUTS
       menuitem - Pointer to first MenuItem in a linked list of
           items.
       vi - Pointer returned by GetVisualInfoA().
       taglist - Pointer to a TagItem list.

   TAGS
       GTMN_TextAttr (struct TextAttr *) - Text Attribute to use for
           menu-items and sub-items.  If not supplied, the screen's
           font will be used.  This font must be openable via OpenFont()
           when this function is called.
       GTMN_Menu (struct Menu *) - Pointer to the Menu structure whose
           FirstItem is the MenuItem supplied above.  If the menu items are
           such that they need to be columnized or shifted, the Menu
           structure is needed to perform the complete calculation.
           It is suggested you always provide this information.

   RESULT
       success - TRUE if successful, false otherwise (signifies that
           the TextAttr wasn't openable).

   EXAMPLE

   NOTES

   BUGS
       If a menu ends up being wider than the whole screen, it will
       run off the right-hand side.

   SEE ALSO
       CreateMenusA(), GetVisualInfoA()
```

gadtools.library

```
gadtools.library/LayoutMenusA                      gadtools.library/LayoutMenusA

   NAME
       LayoutMenusA -- Position all the menus and menu items. (V36)
       LayoutMenus -- Varargs stub for LayoutMenusA(). (V36)

   SYNOPSIS
       success = LayoutMenusA(menu, vi, taglist)
       D0                     A0    A1  A2

       BOOL LayoutMenusA(struct Menu *, APTR, struct TagItem *);

       success = LayoutMenus(menu, vi, firsttag, ...)

       BOOL LayoutMenus(struct Menu *, APTR, Tag, ...);

   FUNCTION
       Lays out all the menus, menu items and sub-items in the supplied
       menu according to the supplied visual information and tag parameters.
       This routine attempts to columnize and/or shift the MenuItems in
       the event that a menu would be too tall or too wide.

   INPUTS
       menu - Pointer to menu obtained from CreateMenusA().
       vi - Pointer returned by GetVisualInfoA().
       taglist - Pointer to a TagItem list.

   TAGS
       GTMN_TextAttr (struct TextAttr *) - Text Attribute to use for
           menu-items and sub-items.  If not supplied, the screen's
           font will be used.  This font must be openable via OpenFont()
           when this function is called.

   RESULT
       success - TRUE if successful, false otherwise (signifies that
           the TextAttr wasn't openable).

   EXAMPLE

   NOTES
       When using this function, there is no need to also call
       LayoutMenuItemsA().

   BUGS
       If a menu ends up being wider than the whole screen, it will
       run off the right-hand side.

   SEE ALSO
       CreateMenusA(), GetVisualInfoA()
```

graphics.library

Page 1

TABLE OF CONTENTS

graphics.library/AddAnimOb
graphics.library/AddBob
graphics.library/AddFont
graphics.library/AddVSprite
graphics.library/AllocRaster
graphics.library/AndRectRegion
graphics.library/AndRegionRegion
graphics.library/Animate
graphics.library/AreaCircle
graphics.library/AreaDraw
graphics.library/AreaEllipse
graphics.library/AreaEnd
graphics.library/AreaMove
graphics.library/AskFont
graphics.library/AskSoftStyle
graphics.library/AttemptLockLayerRom
graphics.library/BitMapScale
graphics.library/BltBitMap
graphics.library/BltBitMapRastPort
graphics.library/BltClear
graphics.library/BltMaskBitMapRastPort
graphics.library/BltPattern
graphics.library/BltTemplate
graphics.library/CBump
graphics.library/CEND
graphics.library/ChangeSprite
graphics.library/CINIT
graphics.library/ClearEOL
graphics.library/ClearRectRegion
graphics.library/ClearRegion
graphics.library/ClearScreen
graphics.library/ClipBlit
graphics.library/CloseFont
graphics.library/CloseMonitor
graphics.library/CMOVE
graphics.library/CopySBitMap
graphics.library/CWAIT
graphics.library/DisownBlitter
graphics.library/DisposeRegion
graphics.library/DoCollision
graphics.library/Draw
graphics.library/DrawEllipse
graphics.library/DrawGList
graphics.library/EraseRect
graphics.library/ExtendFont
graphics.library/FindDisplayInfo
graphics.library/Flood
graphics.library/FontExtent
graphics.library/FreeColorMap
graphics.library/FreeCopList
graphics.library/FreeCprList
graphics.library/FreeGBuffers
graphics.library/FreeRaster
graphics.library/FreeSprite
graphics.library/FreeVPortCopLists
graphics.library/GetColorMap
graphics.library/GetDisplayInfoData
graphics.library/GetGBuffers
graphics.library/GetRGB4
graphics.library/GetSprite
graphics.library/GetVPModeID
graphics.library/GfxAssociate
graphics.library/GfxFree
graphics.library/GfxLookUP

graphics.library

Page 2

graphics.library/GfxNew
graphics.library/InitArea
graphics.library/InitBitMap
graphics.library/InitGels
graphics.library/InitGMasks
graphics.library/InitMasks
graphics.library/InitRastPort
graphics.library/InitTmpRas
graphics.library/InitView
graphics.library/InitVPort
graphics.library/LoadRGB4
graphics.library/LoadView
graphics.library/LockLayerRom
graphics.library/MakeVPort
graphics.library/ModeNotAvailable
graphics.library/Move
graphics.library/MoveSprite
graphics.library/MrgCop
graphics.library/NewRegion
graphics.library/NextDisplayInfo
graphics.library/OpenFont
graphics.library/OpenMonitor
graphics.library/OrRectRegion
graphics.library/OrRegionRegion
graphics.library/OwnBlitter
graphics.library/PolyDraw
graphics.library/QBlit
graphics.library/QBSBlit
graphics.library/ReadPixel
graphics.library/ReadPixelArray8
graphics.library/ReadPixelLine8
graphics.library/RectFill
graphics.library/RemBob
graphics.library/RemFont
graphics.library/RemIBob
graphics.library/RemVSprite
graphics.library/ScalerDiv
graphics.library/ScrollRaster
graphics.library/ScrollVPort
graphics.library/SetAPen
graphics.library/SetBPen
graphics.library/SetCollision
graphics.library/SetDrMd
graphics.library/SetFont
graphics.library/SetOPen
graphics.library/SetRast
graphics.library/SetRGB4
graphics.library/SetRGB4CM
graphics.library/SetSoftStyle
graphics.library/SortGList
graphics.library/StripFont
graphics.library/SyncSBitMap
graphics.library/Text
graphics.library/TextExtent
graphics.library/TextFit
graphics.library/TextLength
graphics.library/UnlockLayerRom
graphics.library/VBeamPos
graphics.library/VideoControl
graphics.library/WaitBlit
graphics.library/WaitBOVP
graphics.library/WaitTOF
graphics.library/WeighTAMatch
graphics.library/WritePixel
graphics.library/WritePixelArray8
graphics.library/WritePixelLine8

```
graphics.library/XorRectRegion
graphics.library/XorRegionRegion
```

```
graphics.library/AddAnimOb                              graphics.library/AddAnimOb

   NAME
        AddAnimOb  --  Add an AnimOb to the linked list of AnimObs.

   SYNOPSIS
        AddAnimOb(anOb, anKey, rp)
                  A0    A1     A2

        void AddAnimOb(struct AnimOb *,struct AnimOb **, struct RastPort *);

   FUNCTION
        Links this AnimOb into the current list pointed to by animKey.
        Initializes all the Timers of the AnimOb's components.
        Calls AddBob with each component's Bob.
        rp->GelsInfo must point to an initialized GelsInfo structure.

   INPUTS
        anOb  = pointer to the AnimOb structure to be added to the list
        anKey = address of a pointer to the first AnimOb in the list
                (anKey = NULL if there are no AnimObs in the list so far)
        rp    = pointer to a valid RastPort

   RESULT

   BUGS

   SEE ALSO
        Animate() graphics/rastport.h graphics/gels.h
```

graphics.library Page 5

```
graphics.library/AddBob                                 graphics.library/AddBob

   NAME
       AddBob -- Adds a Bob to current gel list.

   SYNOPSIS
       AddBob(Bob, rp)
              A0   A1

       void AddBob(struct Bob *, struct RastPort *);

   FUNCTION
       Sets up the system Bob flags, then links this gel into the list
       via AddVSprite.

   INPUTS
       Bob = pointer to the Bob structure to be added to the gel list
       rp  = pointer to a RastPort structure

   RESULT

   BUGS

   SEE ALSO
       InitGels()  AddVSprite()  graphics/gels.h  graphics/rastport.h
```

graphics.library Page 6

```
graphics.library/AddFont                                graphics.library/AddFont

   NAME
       AddFont -- add a font to the system list

   SYNOPSIS
       AddFont(textFont)
               A1

       void AddFont(struct TextFont *);

   FUNCTION
       This function adds the text font to the system, making it
       available for use by any application.  The font added must be
       in public memory, and remain until successfully removed.

   INPUTS
       textFont - a TextFont structure in public ram.

   RESULT

   BUGS

   SEE ALSO
       SetFont()  RemFont()  graphics/text.h
```

graphics.library

Page 7

```
graphics.library/AddVSprite                    graphics.library/AddVSprite

   NAME
        AddVSprite -- Add a VSprite to the current gel list.

   SYNOPSIS
        AddVSprite(vs, rp)
                   A0  A1

        void AddVSprite(struct VSprite *, struct RastPort *);

   FUNCTION
        Sets up the system VSprite flags
        Links this VSprite into the current gel list using its Y,X

   INPUTS
        vs = pointer to the VSprite structure to be added to the gel list
        rp = pointer to a RastPort structure

   RESULT

   BUGS

   SEE ALSO
        InitGels()  graphics/rastport.h  graphics/gels.h
```

graphics.library

Page 8

```
graphics.library/AllocRaster                  graphics.library/AllocRaster

   NAME
       AllocRaster -- Allocate space for a bitplane.

   SYNOPSIS
       planeptr = AllocRaster( width, height )
          d0                    d0:16  d1:16

        PLANEPTR AllocRaster(UWORD,UWORD);

   FUNCTION
       This function calls the memory allocation routines
       to allocate memory space for a bitplane width bits
        wide and height bits high.

   INPUTS
        width   - number of bits wide for bitplane
       height   - number of rows in bitplane

   RESULT
        planeptr - pointer to first word in bitplane, or NULL if
                   it was not possible to allocate the desired
                   amount of memory.
   BUGS

   SEE ALSO
       FreeRaster() graphics/gfx.h
```

```
graphics.library/AndRectRegion                     graphics.library/AndRectRegion

    NAME
        AndRectRegion -- Perform 2d AND operation of rectangle
                         with region, leaving result in region.

    SYNOPSIS
        AndRectRegion(region,rectangle)
                         a0      a1

        void AndRectRegion( struct Region *, struct Rectangle * );

    FUNCTION
        Clip away any portion of the region that exists outside
        of the rectangle. Leave the result in region.

    INPUTS
        region - pointer to Region structure
        rectangle - pointer to Rectangle structure

    NOTES
        Unlike the other rect-region primitives, AndRectRegion() cannot
        fail.

    BUGS

    SEE ALSO
        AndRegionRegion() OrRectRegion() graphics/regions.h
```

```
graphics.library/AndRegionRegion                 graphics.library/AndRegionRegion

    NAME
        AndRegionRegion -- Perform 2d AND operation of one region
                           with second region, leaving result in second region.

    SYNOPSIS
        status = AndRegionRegion(region1,region2)
          d0                       a0      a1

         BOOL AndregionRegion(struct Region *, struct Region * );

    FUNCTION
        Remove any portion of region2 that is not in region1.

    INPUTS
        region1 - pointer to Region structure
        region2 - pointer to Region structure to use and for result

    RESULTS
         status - return TRUE if successful operation
                  return FALSE if ran out of memory

    BUGS

    SEE ALSO
         OrRegionRegion() AndRectRegion() graphics/regions.h
```

graphics.library Page 11

```
graphics.library/Animate                          graphics.library/Animate

   NAME
       Animate  --  Processes every AnimOb in the current animation list.

   SYNOPSIS
       Animate(anKey, rp)
               A0     A1

       void Animate(struct AnimOb **, struct RastPort *);

   FUNCTION
       For every AnimOb in the list
           - update its location and velocities
           - call the AnimOb's special routine if one is supplied
           - for each component of the AnimOb
               - if this sequence times out, switch to the new one
               - call this component's special routine if one is supplied
               - set the sequence's VSprite's y,x coordinates based
                 on whatever these routines cause

   INPUTS
       ankey = address of the variable that points to the head AnimOb
       rp    = pointer to the RastPort structure

   RESULT

   BUGS

   SEE ALSO
       AddAnimOb() graphics/gels.h graphics/rastport.h
```

graphics.library Page 12

```
graphics.library/AreaCircle                    graphics.library/AreaCircle

    NAME
        AreaCircle -- add a circle to areainfo list for areafill.


    SYNOPSIS
        error = (int) AreaCircle( rp,  cx,  cy, radius)
        D0                        A1   D0   D1  D2

        ULONG AreaCircle(struct RastPort *, WORD, WORD, UWORD);

    FUNCTION
        Add circle to the vector buffer. It will be drawn to the rastport when
        AreaEnd is executed.

    INPUTS
        rp       - pointer to a RastPort structure

        cx, cy   - the coordinates of the center of the desired circle.

        radius   - is the radius of the circle to draw around the centerpoint.

    RESULTS
        0 if no error
        -1 if no space left in vector list

    NOTES
        This function is actually a macro which calls
            AreaEllipse(rp,cx,cy,radius,radius).

    SEE ALSO
        AreaMove() AreaDraw() AreaCircle() InitArea() AreaEnd()
        graphics/rastport.h graphics/gfxmacros.h
```

```
graphics.library/AreaDraw                           graphics.library/AreaDraw

   NAME
       AreaDraw -- Add a point to a list of end points for areafill.


   SYNOPSIS
       error = AreaDraw( rp,  x,     y)
         d0              A1 D0:16 D1:16

       ULONG AreaDraw( struct RastPort *, SHORT, SHORT);

   FUNCTION
       Add point to the vector buffer.


   INPUTS
       rp      - points to a RastPort structure.
       x,y     - are coordinates of a point in the raster.

   RESULT
       error   - zero for success, else -1 if no there was no space
                 left in the vector list.

   BUGS

   SEE ALSO
       AreaMove() InitArea() AreaEnd() graphics/rastport.h
```

```
graphics.library/AreaEllipse                     graphics.library/AreaEllipse

   NAME
      AreaEllipse -- add a ellipse to areainfo list for areafill.


   SYNOPSIS
      error = AreaEllipse( rp, cx,   cy,   a,    b    )
      d0                   a1  d0:16 d1:16 d2:16 d3:16

      LONG AreaEllipse( struct RastPort *, SHORT, SHORT, SHORT, SHORT)

   FUNCTION
      Add an ellipse to the vector buffer. It will be draw when AreaEnd() is
      called.

   INPUTS
      rp - pointer to a RastPort structure
      cx - x coordinate of the centerpoint relative to the rastport.
      cy - y coordinate of the centerpoint relative to the rastport.
      a  - the horizontal radius of the ellipse (note: a must be > 0)
      b  - the vertical radius of the ellipse (note: b must be > 0)

   RESULT
      error - zero for success, or -1 if there is no space left in the
              vector list

   SEE ALSO
      AreaMove() AreaDraw() AreaCircle() InitArea() AreaEnd()
      graphics/rastport.h
```

```
graphics.library/AreaEnd                          graphics.library/AreaEnd

   NAME
        AreaEnd -- Process table of vectors and ellipses and produce areafill.


   SYNOPSIS
        error = AreaEnd(rp)
          d0             A1

        LONG AreaEnd( struct RastPort * );

   FUNCTION
        Trigger the filling operation.
        Process the vector buffer and generate required
        fill into the raster planes. After the fill is complete, reinitialize
        for the next AreaMove or AreaEllipse. Use the raster set up by
        InitTmpRas when generating an areafill mask.

   RESULT
        error - zero for success, or -1 if an error occured anywhere.

   INPUTS
        rp - pointer to a RastPort structure which specifies where the filled
             regions will be rendered to.

   BUGS

   SEE ALSO
        InitArea() AreaMove() AreaDraw() AreaEllipse()  InitTmpRas()
        graphics/rastport.h
```

```
graphics.library/AreaMove                        graphics.library/AreaMove

   NAME
        AreaMove -- Define a new starting point for a new
                    shape in the vector list.


   SYNOPSIS
        error =  AreaMove( rp,   x,     y)
         d0                 a1  d0:16  d1:16

        LONG AreaMove( struct RastPort *, SHORT, SHORT );

   FUNCTION
        Close  the last polygon and start another polygon
        at  (x,y). Add the necessary  points  to  vector
        buffer. Closing a polygon may result in the generation
        of another AreaDraw() to close previous polygon.
        Remember to have an initialized AreaInfo structure attached
        to the RastPort.

   INPUTS
        rp  - points to a RastPort structure
        x,y - positions in the raster

   RETURNS
        error - zero for success, or -1 if there is no space left in the
        vector list

   BUGS

   SEE ALSO
        InitArea() AreaDraw() AreaEllipse() AreaEnd() graphics/rastport.h
```

```
graphics.library/AskFont                               graphics.library/AskFont

   NAME
        AskFont -- get the text attributes of the current font

   SYNOPSIS
        AskFont(rp, textAttr)
                A1  A0

        void AskFont(struct RastPort *, struct TextAttr *);

   FUNCTION
        This function fills the text attributes structure with the
        attributes of the current font in the RastPort.

   INPUTS
        rp       - the RastPort from which the text attributes are
                   extracted
        textAttr - the TextAttr structure to be filled.  Note that
                   there is no support for a TTextAttr.

   RESULT
        The textAttr structure is filled with the RastPort's text
        attributes.

   BUGS

   SEE ALSO
        graphics/text.h
```

```
graphics.library/AskSoftStyle                     graphics.library/AskSoftStyle

   NAME
        AskSoftStyle -- Get the soft style bits of the current font.

   SYNOPSIS
        enable = AskSoftStyle(rp)
        D0                    A1

        ULONG AskSoftStyle(struct RastPort *);

   FUNCTION
        This function returns those style bits of the current font
        that are not intrinsic in the font itself, but
        algorithmically generated.  These are the bits that are
        valid to set in the enable mask for SetSoftStyle().

   INPUTS
        rp - the RastPort from which the font and style are extracted.

   RESULTS
        enable - those bits in the style algorithmically generated.
                 Style bits that are not defined are also set.

   BUGS

   SEE ALSO
        SetSoftStyle()  graphics/text.h
```

```
graphics.library/AttemptLockLayerRom     graphics.library/AttemptLockLayerRom
                                      *
   NAME
        AttemptLockLayerRom -- Attempt to Lock Layer structure
                                        by rom(gfx lib) code

   SYNOPSIS
        gotit = AttemptLockLayerRom( layer )
         d0                           a5

        BOOL AttempLockLayerRom(struct Layer *);

   FUNCTION
        Query the current state of the lock on this Layer. If it is
        already locked then return FALSE, could not lock. If the
        Layer was not locked then lock it and return TRUE.
        This call does not destroy any registers.
        This call nests so that callers in this chain will not lock
        themselves out.

   INPUTS
        layer - pointer to Layer structure

   RESULT
        gotit - TRUE or FALSE depending on whether the Layer was
                successfully locked by the caller.

   SEE ALSO
        LockLayerRom() UnlockLayerRom()
```

```
graphics.library/BitMapScale                          graphics.library/BitMapScale

   NAME
        BitMapScale -- Perform raster scaling on a bit map. (V36)

   SYNOPSIS
        BitMapScale(bitScaleArgs)
                    A0

        void BitMapScale(struct BitScaleArgs *);

   FUNCTION
        Scale a source bit map to a non-overlapping destination
        bit map.

   INPUTS
        bitScaleArgs - structure of parameters describing scale:
            bsa_SrcX, bsa_SrcY - origin of the source bits.
            bsa_SrcWidth, bsa_SrcHeight - number of bits to scale from in x
            and y.
            bsa_DestX, bsa_DestY - origin of the destination.
            bsa_DestWidth, bsa_DestHeight - resulting number of bits in x
            and y.  NOTE: these values are set by this function.
            bsa_XSrcFactor:bsa_XDestFactor - equivalant to the ratio
                srcWidth:destWidth, but not necessarily the same
                numbers.  Each must be in the range 1..16383.
            bsa_YSrcFactor:bsa_YDestFactor - equivalant to the ratio
                srcHeight:destHeight, but not necessarily the same
                numbers.  Each must be in the range 1..16383.
            bsa_SrcBitMap - source of the bits to scale.
            bsa_DestBitMap - destination for the bits to scale.  This had
                better be big enough!
            bsa_Flags - future scaling options.  Set it to zero!
            bsa_XDDA, bsa_YDDA - for future use.  Need not be set by user.
            bsa_Reserved1, bsa_Reserved2 - for future use.  Need not be set.

   RESULT
        The destWidth, destHeight fields are set by this function as
        described above.

   NOTES
        o   This function may use the blitter.
        o   Overlapping source and destination bit maps are not
            supported.
        o   No check is made to ensure destBitMap is big enough: use
            ScalerDiv to calculate a destination dimension.

   SEE ALSO
        ScalerDiv()  graphics/scale.h
```

```
graphics.library/BltBitMap                          graphics.library/BltBitMap

   NAME
       BltBitMap -- Move a rectangular region of bits in a BitMap.

   SYNOPSIS
       planecnt = BltBitMap(SrcBitMap, SrcX, SrcY, DstBitMap,
       D0                   A0         D0:16 D1:16 A1
           DstX, DstY, SizeX, SizeY, Minterm, Mask [, TempA])
           D2:16 D3:16 D4:16  D5:16  D6:8     D7:8   [A2]

       ULONG BltBitMap(struct BitMap *, WORD, WORD, struct BitMap *,
           WORD, WORD, WORD, WORD, UBYTE, UBYTE, UWORD *);

   FUNCTION
       Perform non-destructive blits to move a rectangle from one
       area in a BitMap to another area, which can be on a different
       BitMap.
       This blit is assumed to be friendly: no error conditions (e.g.
       a rectangle outside the BitMap bounds) are tested or reported.

   INPUTS
       SrcBitMap, DstBitMap - the BitMap(s) containing the
             rectangles
           - the planes copied from the source to the destination are
             only those whose plane numbers are identical and less
             than the minimum Depth of either BitMap and whose Mask
             bit for that plane is non-zero.
           - as a special case, if a plane pointer in the SrcBitMap
             is zero, it acts as a pointer to a plane of all zeros, and
             if the plane pointer is 0xffffffff, it acts as a pointer
             to a plane of all ones.  (Note: new for V36)
           - SrcBitMap and DstBitMap can be identical if they point
             to actual planes.
       SrcX, SrcY - the x and y coordinates of the upper left corner
           of the source rectangle.  Valid range is positive
           signed integer such that the raster word's offset
           0..(32767-Size)
       DstX, DstY - the x and y coordinates of the upper left
           corner of the destination for the rectangle.  Valid
           range is as for Src.
       SizeX, SizeY - the size of the rectangle to be moved.  Valid
           range is (X: 1..976; Y: 1..1023 such that final raster
           word's offset is 0..32767)
       Minterm - the logic function to apply to the rectangle when
           A is non-zero (i.e. within the rectangle).  B is the
           source rectangle and C, D is the destination for the
           rectangle.
           - $0C0 is a vanilla copy
           - $030 inverts the source before the copy
           - $050 ignores the source and inverts the destination
           - see the hardware reference manual for other combinations
       Mask - the write mask to apply to this operation.  Bits set
           indicate the corresponding planes (if not greater than
           the minimum plane count) are to participate in the
           operation.  Typically this is set to 0xff.
       TempA - If the copy overlaps exactly to the left or right
           (i.e. the scan line addresses overlap), and TempA is
           non-zero, it points to enough chip accessable memory
           to hold a line of A source for the blit (ie CHIP RAM).
           BitBitMap will allocate (and free) the needed TempA if
           none is provided and one is needed.  Blit overlap is
           determined from the relation of the first non-masked
           planes in the source and destination bit maps.

   RESULTS
```

```
       planecnt - the number of planes actually involved in the blit.

   NOTES
       o   This function may use the blitter.

   SEE ALSO
       ClipBlit()  graphics/gfx.h  hardware/blit.h
```

```
graphics.library/BltBitMapRastPort          graphics.library/BltBitMapRastPort

   NAME
        BltBitMapRastPort -- Blit from source bitmap to destination rastport.

   SYNOPSIS
        error = BltBitMapRastPort
                (srcbm, srcx, srcy, destrp, destX, destY, sizeX, sizeY, minterm)
        D0       A0     D0    D1    A1      D2     D3     D4     D5     D6

        BOOL BltBitMapRastPort
             (struct BitMap *, WORD, WORD, struct RastPort *, WORD, WORD,
              WORD, WORD, UBYTE);

   FUNCTION
        Blits from source bitmap to position specified in destination rastport
        using minterm.

   INPUTS
        srcbm   - a pointer to the source bitmap
        srcx    - x offset into source bitmap
        srcy    - y offset into source bitmap
        destrp  - a pointer to the destination rastport
        destX   - x offset into dest rastport
        destY   - y offset into dest rastport
        sizeX   - width of blit in pixels
        sizeY   - height of blit in rows
        minterm - minterm to use for this blit

   RESULT
        TRUE

   BUGS

   SEE ALSO
        BltMaskBitMapRastPort() graphics/gfx.h graphics/rastport.h
```

```
graphics.library/BltClear                                 graphics.library/BltClear

   NAME

        BltClear - Clear a block of memory words to zero.

   SYNOPSIS
        BltClear( memBlock, bytecount, flags )
                  a1        d0         d1

        void BltClear( void *, ULONG, ULONG);

   FUNCTION
        For memory that is local and blitter accessable, the most
        efficient way to clear a range of memory locations is
       to use the system's most efficient data mover, the blitter.
       This command accepts the starting location and count and clears
       that block to zeros.

   INPUTS
        memBloc - pointer to local memory to be cleared
                  memBlock is assumed to be even.
        flags   - set bit 0 to force function to wait until
                  the blit is done.
                  set bit 1 to use row/bytesperrow.

        bytecount       if (flags & 2) == 0 then
                                even number of bytes to clear.
                        else
                                low 16 bits is taken as number of bytes
                                per row and upper 16 bits taken as
                                number of rows.

        This function is somewhat hardware dependant. In the rows/bytesperrow
        mode (with the pre-ECS blitter) rows must be <- 1024. In bytecount mode
        multiple runs of the blitter may be used to clear all the memory.

        Set bit 2 to use the upper 16 bits of the Flags as the data to fill
        memory with instead of 0 (V36).

   RESULT
        The block of memory is initialized.

   BUGS

   SEE ALSO
```

```
graphics.library/BltMaskBitMapRastPort graphics.library/BltMaskBitMapRastPort

   NAME
        BltMaskBitMapRastPort -- blit from source bitmap to destination rastport
        with masking of source image.

   SYNOPSIS
        BltMaskBitMapRastPort
            (srcbm, srcx, srcy, destrp, destX, destY, sizeX, sizeY,
             A0     D0    D1    A1      D2     D3     D4     D5
             minterm, bltmask)
             D6       A2

        void BltMaskBitMapRastPort
             (struct BitMap *, WORD, WORD, struct RastPort *, WORD, WORD,
              WORD, WORD, UBYTE, APTR);

   FUNCTION
        Blits from source bitmap to position specified in destination rastport
        using bltmask to determine where source overlays destination, and
        minterm to determine whether to copy the source image "as is" or
        to "invert" the sense of the source image when copying. In either
        case, blit only occurs where the mask is non-zero.

   INPUTS
        srcbm   - a pointer to the source bitmap
        srcx    - x offset into source bitmap
        srcy    - y offset into source bitmap
        destrp  - a pointer to the destination rastport
        destX   - x offset into dest rastport
        destY   - y offset into dest rastport
        sizeX   - width of blit in pixels
        sizeY   - height of blit in rows
        minterm - either (ABC|ABNC|ANBC) if copy source and blit thru mask
                  or     (ANBC)         if invert source and blit thru mask
        bltmask - pointer to the single bit-plane mask, which must be the
                  same size and dimensions as the planes of the
                  source bitmap.

   RESULT

   BUGS

   SEE ALSO
        BltBitMapRastPort() graphics/gfx.h graphics/rastport.h
```

```
graphics.library/BltPattern                        graphics.library/BltPattern

   NAME
        BltPattern --  Using standard drawing rules for areafill,
                                     blit through a mask.

   SYNOPSIS
       BltPattern(rp, mask, xl, yl, maxx, maxy, bytecnt)
                  a1,  a0   d0  d1   d2   d3     d4

        void BltPattern
           (struct RastPort *, void *, SHORT, SHORT, SHORT, SHORT, SHORT);

   FUNCTION
       Blit using drawmode,areafill pattern, and mask
       at position rectangle (xl,yl) (maxx,maxy).

   INPUTS
       rp     -  points to the destination RastPort for the blit.
       mask   -  points to 2 dimensional mask if needed
                 if mask == NULL then use a rectangle.
       xl,yl -  coordinates of upper left of rectangular region in RastPort
       maxx,maxy - points to lower right of rectangular region in RastPort
       bytecnt - BytesPerRow for mask

   RESULT

   SEE ALSO
        AreaEnd
```

```
graphics.library/BltTemplate                         graphics.library/BltTemplate

   NAME
       BltTemplate -- Cookie cut a shape in a rectangle to the RastPort.

   SYNOPSIS
       BltTemplate(SrcTemplate, SrcX, SrcMod, rp,
                   A0           D0:16  D1:16  A1
           DstX,  DstY, SizeX, SizeY)
           D2:16  D3:16 D4:16  D5:16

       void BltTemplate(UWORD *, WORD, WORD, struct RastPort *,
           WORD, WORD, WORD, WORD);

   FUNCTION
       This function draws the image in the template into the
       RastPort in the current color and drawing mode at the
       specified position.  The template is assumed not to overlap
       the destination.
       If the template falls outside the RastPort boundary, it is
       truncated to that boundary.

       Note: the SrcTemplate pointer should point to the "nearest" word
          (rounded down) of the template mask. Fine alignment of the mask
          is acheived by setting the SrcX bit offseet within the range
          of 0 to 15 decimal.

   INPUTS
       SrcTemplate  - pointer to the first (nearest) word of the template mask.
       SrcX         - x bit offset into the template mask (range 0..15).
       SrcMod       - number of bytes per row in template mask.
       rp           - pointer to destination RastPort.
       DstX, DstY   - x and y coordinates of the upper left
                      corner of the destination for the blit.
       SizeX, SizeY - size of the rectangle to be used as the
                      template.

   NOTES
       o   This function may use the blitter.

   SEE ALSO
       BltBitMap()  graphics/rastport.h
```

```
graphics.library/CBump                                   graphics.library/CBump
   NAME
       CBump -  increment user copper list pointer (bump to next position in list)


   SYNOPSIS
       CBump( c )
              a1

       void CBump( struct UCopList * );

   FUNCTION
       Increment pointer to space for next instruction in user copper list.

   INPUTS
       c - pointer to UCopList structure

   RESULTS
       User copper list pointer is incremented to next position.
       Pointer is repositioned to next user copperlist instruction block
       if the current block is full.

           Note: CBump is usually invoked for the programmer as part of the
                 macro definitions CWAIT or CMOVE.

   BUGS

   SEE ALSO
       CINIT CWAIT CMOVE CEND graphics/copper.h
```

```
graphics.library/CEND                                   graphics.library/CEND
   NAME
       CEND -- Terminate user copper list.

   SYNOPSIS
       CEND( c )

       struct UCopList *c;

   FUNCTION
       Add instruction to terminate user copper list.

   INPUTS
       c - pointer to UCopList structure

   RESULTS
       This is actually a macro that calls the macro CWAIT(c,10000,255)
       10000 is a magical number that the graphics.library uses.
       I hope display technology doesn't catch up too fast!

   BUGS

   SEE ALSO
       CINIT CWAIT CMOVE graphics/copper.h
```

```
graphics.library/ChangeSprite                   graphics.library/ChangeSprite

   NAME
      ChangeSprite -- Change the sprite image pointer.

   SYNOPSIS
      ChangeSprite( vp, s, newdata)
                    a0  a1   a2

       void ChangeSprite(struct ViewPort *, struct SimpleSprite *, void * )

   FUNCTION
       The sprite image is changed to use the data starting at newdata

   INPUTS
      vp - pointer to ViewPort structure that this sprite is
                relative to,  or 0 if relative only top of View
       s - pointer to SimpleSprite structure
       newdata - pointer to data structure of the following form.
               struct spriteimage
               {
                   UWORD    posctl[2]; /* used by simple sprite machine*/
                   UWORD    data[height][2];   /* actual sprite image */
                   UWORD    reserved[2];       /* initialized to */
                                                    /*  0x0,0x0 */
               };
       The programmer must initialize reserved[2].  Spriteimage must be
       in CHIP memory. The height subfield of the SimpleSprite structure
       must be set to reflect the height of the new spriteimage BEFORE
       calling ChangeSprite(). The programmer may allocate two sprites to
       handle a single attached sprite.  After GetSprite(), ChangeSprite(),
       the programmer can set the SPRITE_ATTACHED bit in posctl[1] of the
       odd numbered sprite.
       If you need more than 8 sprites, look up VSprites in the
       graphics documentation.

   RESULTS

   BUGS

   SEE ALSO
       FreeSprite() ChangeSprite() MoveSprite() AddVSprite() graphics/sprite.h
```

```
graphics.library/CINIT                                     graphics.library/CINIT

   NAME
        CINIT -- Initialize user copperlist to accept intermediate
                 user copper instructions.

   SYNOPSIS
        cl = CINIT( ucl , n )

        cl = UCopperListInit( ucl , n )
                              a0    d0

        struct CopList *UCopperListInit( struct UCopList *, UWORD );

   FUNCTION
        Allocates and/or initialize copperlist structures/buffers
        internal to a UCopList structure.

        This is a macro that calls UCopListInit. You must pass a
        (non-initialized) UCopList to CINIT (CINIT will NOT allocate
        a new UCopList if ucl==0 ). If (ucl != 0) it will initialize the
        intermediate data buffers internal to a UCopList.

        The maximum number of intermediate copper list instructions
        that these internal CopList data buffers contain is specified
        as the parameter n.

   INPUTS
        ucl - pointer to UCopList structure
        n - number of instructions buffer must be able to hold

   RESULTS
        cl- a pointer to a buffer which will accept n intermediate copper
            instructions.

        NOTE: this is NOT a UCopList pointer, rather a pointer to the
              UCopList's->FirstCopList sub-structure.

   BUGS
        CINIT will not actually allocate a new UCopList if ucl==0.
        Instead you must allocate a block MEMF_PUBLIC|MEMF_CLEAR, the
        sizeof(struct UCopList) and pass it to this function.

        The system's FreeVPortCopLists function will take care of
        deallocating it if they are called.

        Prior to release V36 the  CINIT macro had { } braces surrounding
        the definition, preventing the proper return of the result value.
        These braces have been removed for the V36 include definitions.

   SEE ALSO
        CINIT CMOVE CEND graphics/copper.h
```

```
graphics.library/ClearEOL                              graphics.library/ClearEOL

   NAME
        ClearEOL -- Clear from current position to end of line.

   SYNOPSIS
        ClearEOL(rp)
                 A1

        void ClearEOL(struct RastPort *);

   FUNCTION
        Clear a rectangular swath from the current position to the
        right edge of the rastPort.  The height of the swath is taken
        from that of the current text font, and the vertical
        positioning of the swath is adjusted by the text baseline,
        such that text output at this position would lie wholly on
        this newly cleared area.
        Clearing consists of setting the color of the swath to zero,
        or, if the DrawMode is 2, to the BgPen.

   INPUTS
        rp - pointer to RastPort structure

   RESULT

   NOTES
        o   This function may use the blitter.

   SEE ALSO
        Text()  ClearScreen()  SetRast()
        graphics/text.h  graphics/rastport.h
```

```
graphics.library/ClearRectRegion            graphics.library/ClearRectRegion

   NAME
        ClearRectRegion -- Perform 2d CLEAR operation of rectangle
                        with region, leaving result in region.

   SYNOPSIS
        status = ClearRectRegion(region,rectangle)
         d0                       a0      a1

        BOOL ClearRectRegion(struct Region *, struct Rectangle * );

   FUNCTION
        Clip away any portion of the region that exists inside
        of the rectangle. Leave the result in region.

   INPUTS
        region - pointer to Region structure
        rectangle - pointer to Rectangle structure

   RESULTS
        status - return TRUE if successful operation
                 return FALSE if ran out of memory

   BUGS

   SEE ALSO
        AndRectRegion() graphics/regions.h
```

```
graphics.library/ClearRegion                       graphics.library/ClearRegion

   NAME
       ClearRegion -- Remove all rectangles from region.

   SYNOPSIS
       ClearRegion(region)
                    a0

        viod ClearRegion( struct Region * );

   FUNCTION
       Clip away all rectangles in the region leaving nothing.

   INPUTS
       region - pointer to Region structure

   BUGS

   SEE ALSO
        NewRegion() graphics/regions.h
```

```
graphics.library/ClearScreen                        graphics.library/ClearScreen

   NAME
       ClearScreen -- Clear from current position to end of RastPort.

   SYNOPSIS
       ClearScreen(rp)
                   A1

       void ClearScreen(struct RastPort *);

   FUNCTION
       Clear a rectangular swath from the current position to the
       right edge of the rastPort with ClearEOL, then clear the rest
       of the screen from just beneath the swath to the bottom of
       the rastPort.
       Clearing consists of setting the color of the swath to zero,
       or, if the DrawMode is 2, to the BgPen.

   INPUTS
       rp - pointer to RastPort structure

   NOTES
       o   This function may use the blitter.

   SEE ALSO
       ClearEOL()  Text()  SetRast()
       graphics/text.h  graphics/rastport.h
```

```
graphics.library/ClipBlit                                  graphics.library/ClipBlit

   NAME
       ClipBlit  --  Calls BltBitMap() after accounting for windows

   SYNOPSIS
       ClipBlit(Src, SrcX, SrcY, Dest, DestX, DestY, XSize, YSize, Minterm)
                A0   D0    D1    A1    D2     D3     D4     D5     D6

       void ClipBlit
            (struct RastPort *, WORD, WORD, struct RastPort *, WORD, WORD,
             WORD, WORD, UBYTE);

   FUNCTION
       Performs the same function as BltBitMap(), except that it
       takes into account the Layers and ClipRects of the layer library,
       all of which are (and should be) transparent to you.  So, whereas
       BltBitMap() requires pointers to BitMaps, ClipBlit requires pointers to
       the RastPorts that contain the Bitmaps, Layers, etcetera.

       If you are going to blit blocks of data around via the RastPort of your
       Intuition Window, you must call this routine (rather than BltBitMap()).

       Either the Src RastPort, the Dest RastPort, both, or neither, can have
       Layers. This routine takes care of all cases.

       See BltBitMap() for a thorough explanation.

   INPUTS
       Src          = pointer to the RastPort of the source for your blit
       SrcX, SrcY   = the topleft offset into Src for your data
       Dest         = pointer to the RastPort to receive the blitted data
       DestX, DestY = the topleft offset into the destination RastPort
       XSize        = the width of the blit
       YSize        = the height of the blit
       Minterm      = the boolean blitter function, where SRCB is associated
                      with the Src RastPort and SRCC goes to the Dest RastPort

   RESULT

   BUGS

   SEE ALSO
       BltBitMap();
```

```
graphics.library/CloseFont                          graphics.library/CloseFont

   NAME
        CloseFont -- Release a pointer to a system font.

   SYNOPSIS
        CloseFont(font)
                  A1

        void CloseFont(struct TextFont *);

   FUNCTION
        This function indicates that the font specified is no longer
        in use.  It is used to close a font opened by OpenFont, so
        that fonts that are no longer in use do not consume system
        resources.

   INPUTS
        font -  a font pointer as returned by OpenFont() or OpenDiskFont()

   RESULT

   BUGS

   SEE ALSO
        OpenFont()  diskfont.library/OpenDiskFont  graphics/text.h
```

```
graphics.library/CloseMonitor                    graphics.library/CloseMonitor

   NAME
       CloseMonitor -- close a MonitorSpec (V36)

   SYNOPSIS
       error = CloseMonitor( monitor_spec )
       d0                    a0

       LONG CloseMonitor( struct MonitorSpec * );

   FUNCTION
       Relinquish access to a MonitorSpec.

   INPUTS
       monitor_spec - a pointer to a MonitorSpec opened via OpenMonitor()

   RESULTS
       error - FALSE if MonitorSpec closed uneventfully.
               TRUE if MonitorSpec could not be closed.

   BUGS

   SEE ALSO
       OpenMonitor()
```

```
graphics.library/CMOVE                                    graphics.library/CMOVE

   NAME
        CMOVE -- append copper move instruction to user copper list.

   SYNOPSIS
        CMOVE( c , a , v )

        CMove( c , a , v )
               a1  d0  d1
        CBump( c )
               a1

        void CMove( struct UCopList *, void *, WORD );

   FUNCTION
        Add instruction to move value v to hardware register a.

   INPUTS
        c - pointer to UCopList structure
        a - hardware register
        v - 16 bit value to be written

   RESULTS
        This is actually a macro that calls CMove(c,&a,v)
        and then calls CBump(c) to bump the local pointer
        to the next instruction. Watch out for macro side affects.

   BUGS

   SEE ALSO
        CINIT CWAIT CEND graphics/copper.h
```

```
graphics.library/CopySBitMap                      graphics.library/CopySBitMap

   NAME
       CopySBitMap --   Syncronize Layer window with contents of
                                             Super BitMap

   SYNOPSIS
       CopySBitMap( layer )
                    a0

        void CopySBitMap(struct Layer *);

   FUNCTION
        This is the inverse of SyncSBitMap.
       Copy all bits from SuperBitMap to Layer bounds.
        This is used for those functions that do not
        want to deal with the ClipRect structures but do want
        to be able to work with a SuperBitMap Layer.

   INPUTS
        layer - pointer to a SuperBitMap Layer
            The Layer must already be locked by the caller.

   BUGS

   SEE ALSO
        LockLayerRom() SyncSBitMap()
```

```
graphics.library/CWAIT                              graphics.library/CWAIT

   NAME
        CWAIT -- Append copper wait instruction to user copper list.

   SYNOPSIS
        CWAIT( c , v , h )

        CWait( c , v , h )
               a1  d0  d1
        CBump( c )
              a1

        void CWait( struct UCopList *, WORD, WORD)

   FUNCTION
        Add instruction to wait for vertical beam position v and
        horizontal position h to this intermediate copper list.

   INPUTS
        c - pointer to UCopList structure
        v - vertical beam position (relative to top of viewport)
        h - horizontal beam position

   RESULTS
        this is actually a macro that calls CWait(c,v,h)
        and then calls CBump(c) to bump the local pointer
        to the next instruction.

   BUGS
        User waiting for horizontal values of greater than 222 decimal
        is illegal.

   SEE ALSO
        CINIT CMOVE CEND graphics/copper.h
```

```
graphics.library/DisownBlitter              graphics.library/DisownBlitter

   NAME
       DisownBlitter - return blitter to free state.


   SYNOPSIS
       DisownBlitter()

        void DisownBlitter( void );

   FUNCTION
        Free blitter up for use by other blitter users.

   INPUTS

   RETURNS

   SEE ALSO
       OwnBlitter()
```

graphics.library Page 43

```
graphics.library/DisposeRegion                  graphics.library/DisposeRegion

   NAME
       DisposeRegion -- Return all space for this region to free
                          memory pool.

   SYNOPSIS
       DisposeRegion(region)
                      a0

        void DisposeRegion( struct Region * );

   FUNCTION
       Free all RegionRectangles for this Region then
        free the Region itself.

   INPUTS
       region - pointer to Region structure

   BUGS

   SEE ALSO
        NewRegion() graphics/regions.h
```

graphics.library Page 44

```
graphics.library/DoCollision                        graphics.library/DoCollision

   NAME
        DoCollision -- Test every gel in gel list for collisions.

   SYNOPSIS
        DoCollision(rp)
                    A1

        void DoCollision(struct RastPort *);

   FUNCTION
        Tests each gel in gel list for boundary and gel-to-gel collisions.
        On detecting one of these collisions, the appropriate collision-
        handling routine is called. See the documentation for a thorough
        description of which collision routine is called. This routine
        expects to find the gel list correctly sorted in Y,X order.
        The system routine SortGList performs this function for the user.

   INPUTS
        rp = pointer to a RastPort

   RESULT

   BUGS

   SEE ALSO
        InitGels()  SortGList()  graphics/gels.h  graphics/gels.h
```

graphics.library Page 45

```
graphics.library/Draw                                    graphics.library/Draw

   NAME
       Draw -- Draw a line between the current pen position
                       and the new x,y position.

   SYNOPSIS
       Draw( rp,   x,     y)
             a1  d0:16  d1:16

        void Draw( struct RastPort *, SHORT, SHORT);

   FUNCTION
       Draw a line from the current pen position to (x,y).

   INPUTS

        rp - pointer to the destination RastPort
        x,y - coordinates of where in the RastPort to end the line.

   BUGS

   SEE ALSO
        Move() graphics/rastport.h
```

graphics.library Page 46

```
graphics.library/DrawEllipse                      graphics.library/DrawEllipse

   NAME
       DrawEllipse -- Draw an ellipse centered at cx,cy with vertical
          and horizontal radii of a,b respectively.

   SYNOPSIS
       DrawEllipse( rp, cx, cy, a, b )
                    a1  d0  d1  d2 d3

       void DrawEllipse( struct RastPort *, SHORT, SHORT, SHORT, SHORT);

   FUNCTION
      Creates an elliptical outline within the rectangular region
       specified by the parameters, using the current foreground pen color.

   INPUTS
       rp - pointer to the RastPort into which the ellipse will be drawn.
       cx - x coordinate of the centerpoint relative to the rastport.
       cy - y coordinate of the centerpoint relative to the rastport.
       a - the horizontal radius of the ellipse (note: a must be > 0)
       b - the vertical radius of the ellipse (note: b must be > 0)

   BUGS

   NOTES
       this routine does not clip the ellipse to a non-layered rastport.

   SEE ALSO
       DrawCircle(), graphics/rastport.h
```

graphics.library Page 47

```
graphics.library/DrawGList                             graphics.library/DrawGList

   NAME
       DrawGList -- Process the gel list, queueing VSprites, drawing Bobs.

   SYNOPSIS
       DrawGList(rp, vp)
                 A1  A0

       void DrawGList(struct RastPort *, struct ViewPort *);

   FUNCTION
       Performs one pass of the current gel list.
          - If nextLine and lastColor are defined, these are
            initialized for each gel.
         - If it's a VSprite, build it into the copper list.
         - If it's a Bob, draw it into the current raster.
         - Copy the save values into the "old" variables,
            double-buffering if required.

   INPUTS
       rp = pointer to the RastPort where Bobs will be drawn
       vp = pointer to the ViewPort for which VSprites will be created

   RESULT

   BUGS
       MUSTDRAW isn't implemented yet.

   SEE ALSO
       InitGels()  graphics/gels.h graphics/rastport.h  graphics/view.h
```

graphics.library Page 48

```
graphics.library/EraseRect                             graphics.library/EraseRect

   NAME

       EraseRect -- Fill a defined rectangular area using the current
                    BackFill hook. (V36)

   SYNOPSIS
       EraseRect( rp, xmin, ymin, xmax, ymax)
                  a1  d0:16 d1:16 d2:16 d3:16

       void EraseRect(struct RastPort *, SHORT, SHORT, SHORT, SHORT);

   FUNCTION
       Fill the rectangular region specified by the parameters with the
       BackFill hook. If non-layered, the rectangular region specified by
       the parameters is cleared. If layered the Layer->BackFill Hook is used.

   INPUTS
       rp      - pointer to a RastPort structure
       xmin    - x coordinate of the upper left corner of the region to fill.
       ymin    - y coordinate of the upper left corner of the region to fill.
       xmax    - x coordinate of the lower right corner of the region to fill.
       ymax    - y coordinate of the lower right corner of the region to fill.

   BUGS

   NOTES
       The following relation MUST be true:
       (xmax >= xmin) and (ymax >= ymin)

   SEE ALSO
       graphics/rastport.h graphics/clip.h
```

```
graphics.library/ExtendFont                    graphics.library/ExtendFont

   NAME
        ExtendFont -- ensure tf_Extension has been built for a font (V36)

   SYNOPSIS
        success = ExtendFont(font, fontTags)
        D0                   A0    A1

        ULONG ExtendFont(struct TextFont *, struct TagItem *);

   SEE ALSO
        graphics/text.h
```

```
graphics.library/FindDisplayInfo          graphics.library/FindDisplayInfo

   NAME
        FindDisplayInfo -- search for a record identified by a specific key (V36)

   SYNOPSIS
        handle = FindDisplayInfo(ID)
        D0                       D0

        DisplayInfoHandle FindDisplayInfo(ULONG);

   FUNCTION
        Given a 32-bit Mode Key, return a handle to a valid DisplayInfoRecord
        found in the graphics database.  Using this handle, you can obtain
        information about this Mode, including its default dimensions,
        properties, and whether it is currently available for use.

   INPUTS
        ID     - unsigned long identifier

   RESULT
        handle - handle to a displayinfo Record with that key
                 or NULL if no match.

   BUGS

   SEE ALSO
        graphics/displayinfo.h
```

```
graphics.library/Flood                                   graphics.library/Flood

   NAME
       Flood -- Flood rastport like areafill.

   SYNOPSIS
       error = Flood( rp, mode, x, y)
       d0             a1   d2   d0  d1

       BOOL Flood(struct RastPort *, ULONG, SHORT, SHORT);

   FUNCTION
       Search the BitMap starting at (x,y).
       Fill all adjacent pixels if they are:
           Mode 0: not the same color as AOLPen
           Mode 1: the same color as the pixel at (x,y)

       When actually doing the fill use the modes that apply to
       standard areafill routine such as drawmodes and patterns.

   INPUTS
       rp - pointer to RastPort
       (x,y) - coordinate in BitMap to start the flood fill at.
       mode -  0 fill all adjacent pixels searching for border.
               1 fill all adjacent pixels that have same pen number
               as the one at (x,y).

   NOTES
       In order to use Flood, the destination RastPort must
       have a valid TmpRas raster whose size is as large as
       that of the RastPort.

   SEE ALSO
       AreaEnd() InitTmpRas() graphics/rastport.h
```

```
graphics.library/FontExtent                         graphics.library/FontExtent

   NAME
       FontExtent -- get the font attributes of the current font (V36)

   SYNOPSIS
       FontExtent(font, fontExtent)
                  A0    A1

       void FontExtent(struct TextFont *, struct TextExtent *);

   FUNCTION
       This function fills the text extent structure with a bounding
       (i.e. maximum) extent for the characters in the specified font.

   INPUTS
       font       - the TextFont from which the font metrics are extracted.
       fontExtent - the TextExtent structure to be filled.

   RESULT
       fontExtent is filled.

   NOTES
       The TextFont, not the RastPort, is specified -- unlike
       TextExtent(), effect of algorithmic enhancements is not
       included, nor does te_Width include any effect of
       rp_TxSpacing.  The returned te_Width will be negative only
       when FPF_REVPATH is set in the tf_Flags of the font -- the
       effect of left-moving characters is ignored for the width of
       a normal font, and the effect of right-moving characters is
       ignored if a REVPATH font.  These characters will, however,
       be reflected in the bounding extent.

   SEE ALSO
       TextExtent()  graphics/text.h
```

```
graphics.library/FreeColorMap                          graphics.library/FreeColorMap

   NAME
       FreeColorMap -- Free the ColorMap structure and return memory
                                               to free memory pool.

   SYNOPSIS
       FreeColorMap( colormap )
                       a0

        void FreeColorMap(struct ColorMap *);

   FUNCTION
        Return the memory to the free memory pool that was allocated
        with GetColorMap.

   INPUTS
        colormap - pointer to ColorMap allocated with GetColorMap

   RESULT
        The space is made available for others to use.

   BUGS

   SEE ALSO
       SetRGB4() GetColorMap() graphics/view.h
```

```
graphics.library/FreeCopList                              graphics.library/FreeCopList

   NAME
        FreeCopList -- deallocate intermediate copper list

   SYNOPSIS
       FreeCopList(coplist)
                   a0

        void FreeCopList( struct CopList *);

   FUNCTION
        Deallocate all memory associated with this copper list.

   INPUTS
       coplist  - pointer to structure CopList

   RESULTS
        memory returned to memory manager

   BUGS

   SEE ALSO
        graphics/copper.h
```

graphics.library Page 55

```
graphics.library/FreeCprList                      graphics.library/FreeCprList

   NAME
       FreeCprList -- deallocate hardware copper list

   SYNOPSIS
       FreeCprList(cprlist)
                   a0

        void FreeCprList(struct cprlist *);

   FUNCTION
       return cprlist to free memory pool

   INPUTS
       cprlist - pointer to cprlist structure

   RESULTS
        memory returned and made available to other tasks

   BUGS

   SEE ALSO
        graphics/copper.h
```

graphics.library Page 56

```
graphics.library/FreeGBuffers                          graphics.library/FreeGBuffers

   NAME
       FreeGBuffers -- Deallocate memory obtained by GetGBufers.

   SYNOPSIS
       FreeGBuffers(anOb, rp, db)
                    A0    A1  D0

       void FreeGBuffers(struct AnimOb *, struct RastPort *, BOOL);

   FUNCTION
       For each sequence of each component of the AnimOb,
       deallocate memory for:
           SaveBuffer
           BorderLine
           CollMask and ImageShadow (point to same buffer)
           if db is set (user had used double-buffering) deallocate:
               DBufPacket
               BufBuffer

   INPUTS
       anOb = pointer to the AnimOb structure
       rp   = pointer to the current RastPort
       db   = double-buffer indicator (set TRUE for double-buffering)

   RESULT

   BUGS

   SEE ALSO
       GetGBuffers()  graphics/gels.h  graphics/rastport.h
```

```
graphics.library/FreeRaster                         graphics.library/FreeRaster

   NAME
       FreeRaster -- Release an allocated area to the system free memory pool


   SYNOPSIS
       FreeRaster( p, width, height)
                   a0   d0:16  d1:16

        void FreeRaster( PLANEPTR, USHORT, USHORT);

   FUNCTION
        Return the memory associated with this PLANEPTR of size
        width and height to the MEMF_CHIP memory pool.

   INPUTS
       p  =  a pointer to a memory space  returned  as  a
             result of a call to AllocRaster.

        width - the width in bits of the bitplane.
        height - number of rows in bitplane.

   BUGS

   NOTES
       Width and height should be the same values with which you
       called AllocRaster in the first place.

   SEE ALSO
        AllocRaster() graphics/gfx.h
```

```
graphics.library/FreeSprite                         graphics.library/FreeSprite

   NAME
       FreeSprite -- Return sprite for use by others and virtual
                                   sprite machine.

   SYNOPSIS
       FreeSprite( pick )
                   d0

        void FreeSprite( WORD );

   FUNCTION
        Mark sprite as available for others to use.
       These sprite routines are provided to ease sharing of sprite
        hardware and to handle simple cases of sprite usage and
        movement.  It is assumed the programs that use these routines
        do want to be good citizens in their hearts. ie: they will
        not FreeSprite unless they actually own the sprite.
        The Virtual Sprite machine may ignore the simple sprite machine.

   INPUTS
       pick - number in range of 0-7

   RESULTS
        sprite made available for subsequent callers of GetSprite
        as well as use by Virtual Sprite Machine.

   BUGS

   SEE ALSO
       GetSprite() ChangeSprite() MoveSprite() graphics/sprite.h
```

```
graphics.library/FreeVPortCopLists         graphics.library/FreeVPortCopLists

   NAME
       FreeVPortCopLists -- deallocate all intermediate copper lists and
       their headers from a viewport

   SYNOPSIS
       FreeVPortCopLists(vp)
                         a0

        void FreeVPortCopLists(struct ViewPort *);

   FUNCTION
       Search display, color, sprite, and user copper
       lists and call FreeMem() to deallocate them from memory

   INPUTS
       vp - pointer to ViewPort structure

   RESULTS
       The memory allocated to the various copper lists will be returned
        to the system's free memory pool, and the following fields in
        the viewport structure will be set to NULL:

                DspIns, Sprins, ClrIns, UCopIns

   BUGS
       none known

   SEE ALSO
        graphics/view.h
```

```
graphics.library/GetColorMap                         graphics.library/GetColorMap

   NAME
       GetColorMap -- allocate and initialize Colormap


   SYNOPSIS
       cm = GetColorMap( entries )
       d0                d0

        struct ColorMap *GetColorMap( ULONG);

   FUNCTION
       Allocates, initializes and returns a pointer to a ColorMap
       data structure, later enabling calls to SetRGB4
       and LoadRGB4 to load colors for a view port. The ColorTable
        pointer in the ColorMap structure points to a hardware
        specific colormap data structure. You should not count on
        it being anything you can understand. Use GetRGB4() to
        query it or SetRGB4CM to set it directly.

   INPUTS
        entries - number of entries for this colormap

    RESULT
        The pointer value returned by this routine, if nonzero,
       may be stored into the ViewPort.ColorMap pointer.
       If a value of 0 is returned, the system was unable
       to allocate enough memory space for the required
       data structures.

   BUGS

   SEE ALSO
       SetRGB4() FreeColorMap()
```

graphics.library

Page 61

```
graphics.library/GetDisplayInfoData       graphics.library/GetDisplayInfoData

   NAME
        GetDisplayInfoData -- query DisplayInfo Record parameters (V36)

   SYNOPSIS
        result = GetDisplayInfoData(handle, buf, size, tagID, [ID])
        D0                          A0      A1   D0    D1     [D2]

        ULONG GetDisplayInfoData(DisplayInfoHandle, UBYTE *, ULONG, ULONG, ULONG);

   FUNCTION
        GetDisplayInfoData() fills a buffer with data meaningful to the
        DisplayInfoRecord pointed at by your valid handle. The data type
        that you are interested in is indicated by a tagID for that chunk.
        The types of tagged information that may be available include:

        DTAG_DISP: (DisplayInfo)   - properties and availability information.
        DTAG_DIMS: (DimensionInfo) - default dimensions and overscan info.
        DTAG_MNTR: (MonitorInfo)   - type, position, scanrate, and compatibility
        DTAG_NAME: (NameInfo)      - a user friendly way to refer to this mode.

   INPUTS
        handle - displayinfo handle
        buf    - pointer to destination buffer
        size   - buffer size in bytes
        tagID  - data chunk type
        ID     - displayinfo identifier, optionally used if handle is NULL

   RESULT
        result - if positive, number of bytes actually transferred
                 if zero, no information for ID was available

   BUGS

   SEE ALSO
        FindDisplayInfo(), NextDisplayInfo()
        graphics/displayinfo.h
```

---

graphics.library

Page 62

```
graphics.library/GetGBuffers                        graphics.library/GetGBuffers

   NAME
        GetGBuffers -- Attempt to allocate ALL buffers of an entire AnimOb.

   SYNOPSIS
        status = GetGBuffers(anOb, rp, db)
        D0                   A0    A1  D0

        BOOL GetGBuffers(struct AnimOb *, struct RastPort *, BOOL);

   FUNCTION
        For each sequence of each component of the AnimOb, allocate memory for:
            SaveBuffer
            BorderLine
            CollMask and ImageShadow (point to same buffer)
            if db is set TRUE (user wants double-buffering) allocate:
                DBufPacket
                BufBuffer

   INPUTS
        anOb = pointer to the AnimOb structure
        rp   = pointer to the current RastPort
        db   = double-buffer indicator (set TRUE for double-buffering)

   RESULT
        status = TRUE if the memory allocations were all successful, else FALSE

   BUGS
        If any of the memory allocations fail it does not free the partial
        allocations that did succeed.

   SEE ALSO
        FreeGBuffers() graphics/gels.h
```

```
graphics.library/GetRGB4                                graphics.library/GetRGB4

   NAME
       GetRGB4 -- Inquire value of entry in ColorMap.

   SYNOPSIS
       value = GetRGB4( colormap, entry )
         d0               a0       d0

        ULONG GetRGB4(struct ColorMap *, LONG);

   FUNCTION
        Read and format a value from the ColorMap.

   INPUTS
        colormap - pointer to ColorMap structure
        entry - index into colormap

   RESULT
        returns -1 if no valid entry
        return UWORD RGB value 4 bits per gun right justified

   BUGS

   SEE ALSO
       SetRGB4() LoadRGB4() GetColorMap() FreeColorMap() graphics/view.h
```

```
graphics.library/GetSprite                          graphics.library/GetSprite

   NAME
        GetSprite -- Attempt to get a sprite for the simple sprite
                                manager.

   SYNOPSIS
        Sprite_Number = GetSprite( sprite, pick )
            d0                      a0      d0

        SHORT GetSprite( struct SimpleSprite *, SHORT );

   FUNCTION
        Attempt to allocate one of the eight sprites for private use
        with the simple sprite manager. This must be done before using
        further calls to the simple sprite machine. If the programmer
        wants to use 15 color sprites, they must allocate both sprites
        and set the 'SPRITE_ATTACHED' bit in the odd sprite's posctldata
        array.

   INPUTS
        sprite - ptr to programmers SimpleSprite structure.
        pick - number in the range of 0-7 or
          -1 if programmer just wants the next one.

   RESULTS
        If pick is 0-7 attempt to allocate the sprite. If the sprite
        is already allocated then return -1.
        If pick -1 allocate the next sprite starting search at 0.
        If no sprites are available return -1 and fill -1 in num entry
        of SimpleSprite structure.
        If the sprite is available for allocation, mark it allocated
        and fill in the 'num' entry of the SimpleSprite structure.
        If successful return the sprite number.

   BUGS

   SEE ALSO
        FreeSprite() ChangeSprite() MoveSprite() GetSprite() graphics/sprite.h
```

graphics.library

Page 65

```
graphics.library/GetVPModeID                        graphics.library/GetVPModeID

   NAME
       GetVPModeID -- get the 32 bit DisplayID from a ViewPort. (V36)

   SYNOPSIS
       modeID =  GetVPModeID( vp )
       d0                     a0

       ULONG GetVPModeID( struct ViewPort *);

   FUNCTION
       returns the normal display modeID, if one is currently  associated
       with this ViewPort.

   INPUTS
       vp -- pointer to a ViewPort structure.

   RESULT

       modeID -- a 32 bit DisplayInfoRecord identifier associated with
                 this ViewPort, or INVALID_ID.

   NOTES
       Test the return value of this function against INVALID_ID, not NULL.
       (INVALID_ID is defined in graphics/displayinfo.h).

   BUGS

   SEE ALSO
       graphics/displayinfo.h, ModeNotAvailable()
```

graphics.library

Page 66

```
graphics.library/GfxAssociate                          graphics.library/GfxAssociate

   NAME
       GfxAssociate -- associate a graphics extended node with a given pointer
                       (V36)

   SYNOPSIS
      GfxAssociate(pointer, node);
                   A0       A1

       void GfxAssociate(VOID *, struct ExtendedNode *);

   FUNCTION
       Associate a special graphics extended data structure (each of which
       begins with an ExtendedNode structure)  with another structure via
       the other structure's pointer. Later, when you call GfxLookUp()
       with the other structure's pointer you may retrieve a pointer
       to this special graphics extended data structure, if it is
       available.

   INPUTS
       pointer = a pointer to a data structure.
       node = an ExtendedNode structure to associate with the pointer

   RESULT
       an association is created between the pointer and the node such
       that given the pointer the node can be retrieved via GfxLookUp().

   BUGS

   SEE ALSO
       graphics/gfxnodes.h GfxNew() GfxFree() GfxLookUp()
```

graphics.library Page 67

```
graphics.library/GfxFree                                    graphics.library/GfxFree

   NAME
      GfxFree -- free a graphics extended data structure (V36)

   SYNOPSIS
      GfxFree( node );
             a0

       void GfxFree(struct ExtendedNode *);

   FUNCTION
       Free a special graphics extended data structure (each of which
       begins with an ExtendedNode structure).

   INPUTS
       node = pointer to a graphics extended data structure obtained via
              GfxNew().

   RESULT
       the node is deallocated from memory. graphics will dissassociate
       this special graphics extended node from any associated data
       structures, if necessary, before freeing it (see GfxAssociate()).

   BUGS
       an Alert() will be called if you attempt to free any structure
       other than a graphics extended data strucure obtained via GfxFree().

   SEE ALSO
       graphics/gfxnodes.h GfxNew() GfxAssociate() GfxLookUp()
```

graphics.library Page 68

```
graphics.library/GfxLookUP                                  graphics.library/GfxLookUP

   NAME
       GfxLookUp -- find a graphics extended node associated with a
                    given pointer (V36)

   SYNOPSIS
      result = GfxLookUp( pointer );
      d0                  a0

       struct ExtendedNode *GfxLookUp( void *);

   FUNCTION
       Finds a special graphics extended data structure (if any) associated
       with the pointer to a data structure (eg: ViewExtra associated with
       a View structure).

   INPUTS
       pointer = a pointer to a data structure which may have an
                 ExtendedNode associated with it (typically a View ).

   RESULT
       result = a pointer to the ExtendedNode that has previously been
                associated with the pointer.

   BUGS

   SEE ALSO
       graphics/gfxnodes.h GfxNew() GfxFree() GfxAssociate()
```

```
graphics.library/GfxNew                                   graphics.library/GfxNew

   NAME
       GfxNew -- allocate a graphics extended data structure (V36)

   SYNOPSIS
        result = GfxNew( node_type );
        d0               d0

        struct ExtendedNode *GfxNew( ULONG);

   FUNCTION
        Allocate a special graphics extended data structure (each of which
        begins with an ExtendedNode structure).  The type of structure to
        be allocated is specified by the node_type identifier.

   INPUTS
        node_type = which type of graphics extended data structure to allocate.
                    (see gfxnodes.h for identifier definitions.)

   RESULT
        result = a pointer to the allocated graphics node or NULL if the
                 allocation failed.

   BUGS

   SEE ALSO
        graphics/gfxnodes.h GfxFree() GfxAssociate() GfxLookUp()
```

```
graphics.library/InitArea                                 graphics.library/InitArea

   NAME

        InitArea -- Initialize vector collection matrix

   SYNOPSIS

        InitArea( areainfo, buffer, maxvectors )
                  a0        a1      d0

        void InitArea(struct AreaInfo *, void *, SHORT);

   FUNCTION
        This function provides initialization for the vector collection matrix
        such that it has a size of (max vectors ).  The size of the region
        pointed to by buffer (short pointer) should be five (5) times as large
        as maxvectors. This size is in bytes.  Areafills done by using AreaMove,
        AreaDraw, and AreaEnd must have enough space allocated in this table to
        store all the points of the largest fill. AreaEllipse takes up two
        vectors for every call. If AreaMove/Draw/Ellipse detect too many
        vectors going into the buffer they will return -1.

   INPUTS
        areainfo - pointer to AreaInfo structure
        buffer - pointer to chunk of memory to collect vertices
        maxvectors - max number of vectors this buffer can hold

   RESULT
        Pointers are set up to begin storage of vectors done by
        AreaMove, AreaDraw, and AreaEllipse.

   BUGS

   SEE ALSO
        AreaEnd() AreaMove() AreaDraw() AreaEllipse() graphics/rastport.h
```

graphics.library

```
graphics.library/InitBitMap                         graphics.library/InitBitMap

   NAME

       InitBitMap -- Initialize bit map structure with input values.

   SYNOPSIS
       InitBitMap( bm, depth, width, height )
                   a0   d0     d1      d2

       void InitBitMap( struct BitMap *, BYTE, UWORD, UWORD );

   FUNCTION
       Initialize various elements in the BitMap structure to
       correctly reflect depth, width, and height.
       Must be used before use of BitMap in other graphics calls.
       The Planes[8] are not initialized and need to be set up
       by the caller.  The Planes table was put at the end of the
       structure so that it may be truncated to conserve space,
       as well as extended. All routines that use BitMap should
       only depend on existence of depth number of bitplanes.
       The Flagsh and pad fields are reserved for future use and
       should not be used by application programs.

   INPUTS
       bm - pointer to a BitMap structure (gfx.h)
       depth - number of bitplanes that this bitmap will have
       width - number of bits (columns) wide for this BitMap
       height- number of bits (rows) tall for this BitMap

   BUGS

   SEE ALSO
       graphics/gfx.h
```

graphics.library

```
graphics.library/InitGels                                 graphics.library/InitGels

   NAME
       InitGels -- initialize a gel list; must be called before using gels.

   SYNOPSIS
       InitGels(head, tail, GInfo)
                 A0    A1    A2

       void InitGels(struct VSprite *, struct VSprite *, struct GelsInfo *);

   FUNCTION
       Assigns the VSprites as the head and tail of the gel list in GfxBase.
       Links these two gels together as the keystones of the list.
       If the collHandler vector points to some memory array, sets
       the BORDERHIT vector to NULL.

   INPUTS
       head  = pointer to the VSprite structure to be used as the gel list head
       tail  = pointer to the VSprite structure to be used as the gel list tail
       GInfo = pointer to the GelsInfo structure to be initialized

   RESULT

   BUGS

   SEE ALSO
       graphics/gels.h  graphics/rastport.h
```

```
graphics.library/InitGMasks                          graphics.library/InitGMasks

   NAME
        InitGMasks -- Initialize all of the masks of an AnimOb.

   SYNOPSIS
        InitGMasks(anOb)
                   A0

        void InitGMasks(struct AnimOb *);

   FUNCTION
        For every sequence of every component call InitMasks.

   INPUTS
        anOb = pointer to the AnimOb

   BUGS

   SEE ALSO
        InitMasks() graphics/gels.h
```

```
graphics.library/InitMasks                                 graphics.library/InitMasks

   NAME
        InitMasks -- Initialize the BorderLine and CollMask masks of a VSprite.

   SYNOPSIS
        InitMasks(vs)
                  A0

        void InitMasks(struct VSprite *);

   FUNCTION
        Creates the appropriate BorderLine and CollMask masks of the VSprite.
        Correctly detects if the VSprite is actually a Bob definition, handles
        the image data accordingly.

   INPUTS
        vs = pointer to the VSprite structure

   RESULT

   BUGS

   SEE ALSO
        InitGels()  graphics/gels.h
```

graphics.library Page 75

```
graphics.library/InitRastPort                    graphics.library/InitRastPort

   NAME
        InitRastPort -- Initialize raster port structure

   SYNOPSIS
        InitRastPort( rp )
                      a1

        struct RastPort *rp;

   FUNCTION
       Initialize a RastPort structure to standard values.

   INPUTS
        rp      = pointer to a RastPort structure.

   RESULT
        all entries in RastPort get zeroed out, with the following exceptions:

            Mask, FgPen, AOLPen, and LinePtrn are set to -1.
            The DrawMode is set to JAM2
            The font is set to the standard system font

   NOTES
        The struct Rastport describes a control structure
       for a write-able raster. The RastPort structure
       describes how a complete single playfield display
       will be written into. A RastPort structure is
       referenced whenever any drawing or filling
       operations are to be performed on a section of
       memory.

       The section of memory which is being used in this
       way may or may not be presently a part of the
       current actual onscreen display memory. The name
       of the actual memory section which is linked to
       the RastPort is referred to here as a "raster" or
       as a bitmap.

       NOTE: Calling the routine InitRastPort only
       establishes various defaults. It does NOT
       establish where, in memory, the rasters are
       located. To do graphics with this RastPort the user
        must set up the BitMap pointer in the RastPort.

   BUGS

   SEE ALSO
        graphics/rastport.h
```

graphics.library Page 76

```
graphics.library/InitTmpRas                          graphics.library/InitTmpRas

   NAME
        InitTmpRas -- Initialize area of local memory for usage by
                        areafill, floodfill, text.

   SYNOPSIS
        InitTmpRas(tmpras, buffer, size)
                   a0       a1     d0

        void InitTmpRas( struct TmpRas *, void *, ULONG );

   FUNCTION
        The area of memory pointed to by buffer is set up to be used
        by RastPort routines that may need to get some memory for
        intermediate operations in preparation to putting the graphics
        into the final BitMap.
        Tmpras is used to control the usage of buffer.

   INPUTS
        tmpras - pointer to a TmpRas structure to be linked into
                a RastPort
        buffer - pointer to a contguous piece of chip memory.
        size - size in bytes of buffer

   RESULT
        makes buffer available for users of RastPort

   BUGS
        Would be nice if RastPorts could share one TmpRas.

   SEE ALSO
        AreaEnd() Flood() Text() graphics/rastport.h
```

```
graphics.library/InitView                                   graphics.library/InitView

   NAME
   InitView - Initialize View structure.

   SYNOPSIS
        InitView( view )
                  a1

        void InitView( struct View * );

   FUNCTION
        Initialize View structure to default values.

   INPUTS
        view - pointer to a View structure

   RESULT
        View structure set to all 0's. (1.0,1.1.1.2)
        Then values are put in DxOffset,DyOffset to properly position
        default display about .5 inches from top and left on monitor.
        InitView pays no attention to previous contents of view.

   BUGS

   SEE ALSO
        MakeVPort graphics/view.h
```

```
graphics.library/InitVPort                                 graphics.library/InitVPort

   NAME
        InitVPort - Initialize ViewPort structure.

   SYNOPSIS
        InitVPort( vp )
                   a0

        void InitViewPort( struct ViewPort * );

   FUNCTION
        Initialize ViewPort structure to default values.

   INPUTS
        vp - pointer to a ViewPort structure

   RESULT
        ViewPort structure set to all 0's. (1.0,1.1)
       New field added SpritePriorities, initialized to 0x24 (1.2)

   BUGS

   SEE ALSO
        MakeVPort() graphics/view.h
```

graphics.library

Page 79

```
graphics.library/LoadRGB4                              graphics.library/LoadRGB4

   NAME
        LoadRGB4 -- Load RGB color values from table.

   SYNOPSIS
        LoadRGB4( vp, colors , count )
                  a0    a1      d0:16

        void LoadRGB4( struct ViewPort *, UWORD *, WORD);

   FUNCTION
        load the count words of the colormap from table starting at
        entry 0.

   INPUTS
        vp - pointer to ViewPort, whose colors you wish to change
        colors - pointer to table of RGB values set up as an array
                 of USHORTS
                        background--  0x0RGB
                        color1    --  0x0RGB
                        color2    --  0x0RGB
                         etc.         UWORD per value.
                The colors are interpreted as 15 = maximum intensity.
                                              0 = minimum intensity.
        count   = number of UWORDs in the table to load into the
          colormap starting at color 0(background) and proceeding
          to the next higher color number

   RESULTS
        The ViewPort should have a pointer to a valid ColorMap to store
        the colors in.
        Updates the hardware copperlist to reflect the new colors.
        Updates the intermediate copperlist with the new colors.

   BUGS

   SEE ALSO
        SetRGB4() GetRGB4() GetColorMap() graphics/view.h
```

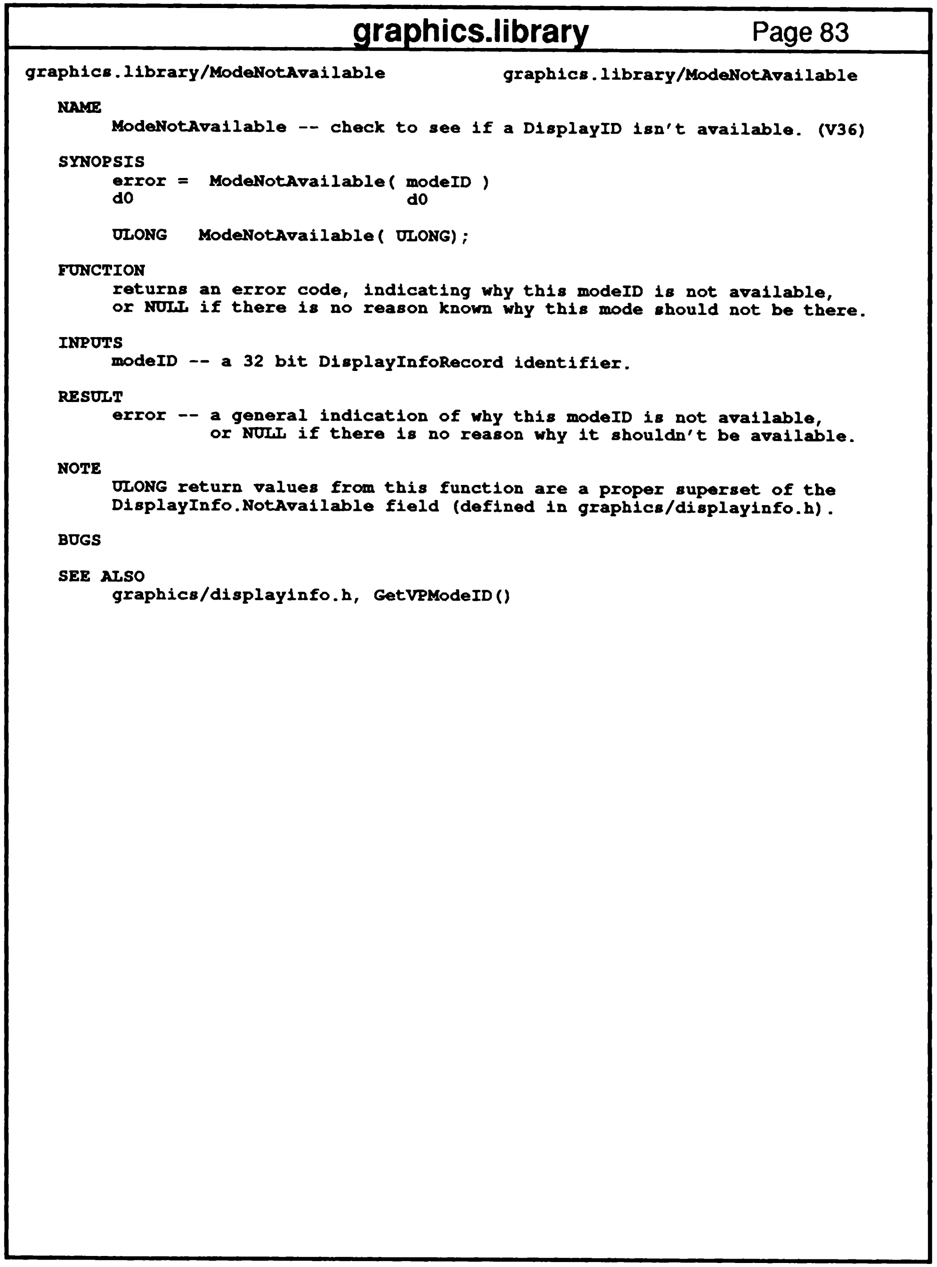
graphics.library

Page 80

```
graphics.library/LoadView                              graphics.library/LoadView

   NAME
       LoadView -- Use a (possibly freshly created) coprocessor instruction
                   list to create the current display.

   SYNOPSIS
       LoadView( View )
                 A1

        void LoadView( struct View * );

   FUNCTION
        Install a new view to be displayed during the next display
        refresh pass.
       Coprocessor instruction list has been created by
       InitVPort(), MakeView(), and MrgCop().

   INPUTS
       View - a pointer to the View structure which contains the
       pointer to the constructed coprocessor instructions list, or NULL.

   RESULT
        If the View pointer is non-NULL, the new View is displayed,
        according to your instructions.  The vertical blank routine
        will pick this pointer up and direct the copper to start
        displaying this View.

        If the View pointer is NULL, no View is displayed.

   NOTE
        Even though a LoadView(NULL) is performed, display DMA will still be
        active.  Sprites will continue to be displayed after a LoadView(NULL)
        unless an OFF_SPRITE is subsequently performed.

   BUGS

   SEE ALSO
       InitVPort() MakeVPort() MrgCop() intuition/RethinkDisplay()
        graphics/view.h
```

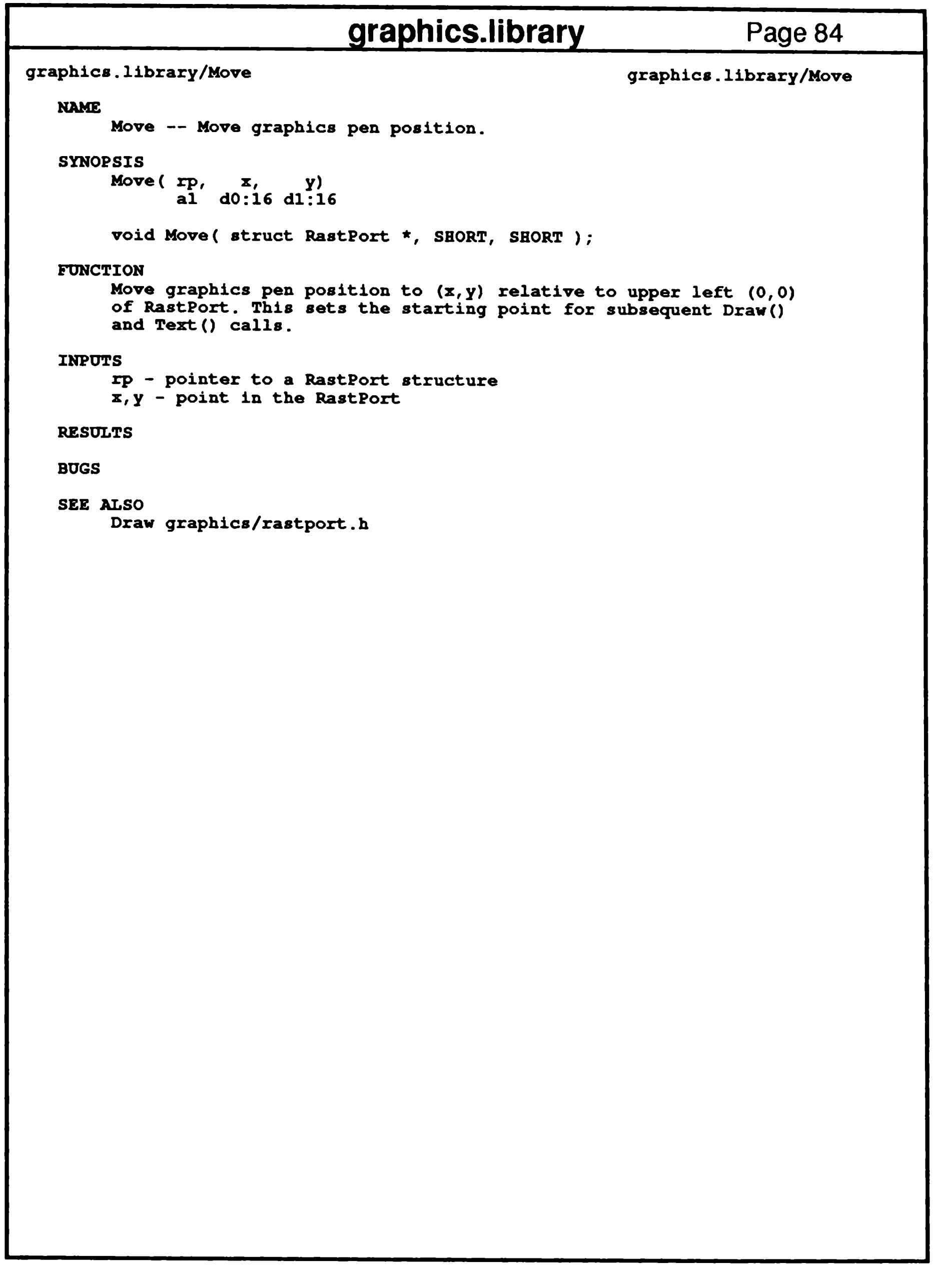
```
graphics.library/LockLayerRom                    graphics.library/LockLayerRom

   NAME
        LockLayerRom -- Lock Layer structure by rom(gfx lib) code.

   SYNOPSIS
        LockLayerRom( layer )
                      a5

        void LockLayerRom( struct Layer * );

   FUNCTION
        Return when the layer is locked and no other task may
        alter the ClipRect structure in the Layer structure.
        This call does not destroy any registers.
        This call nests so that callers in this chain will not lock
        themselves out.
        Do not have the Layer locked during a call to intuition.
        There is a potential deadlock problem here, if intuition
        needs to get other locks as well.
        Having the layer locked prevents other tasks from using the
        layer library functions, most notably intuition itself. So
        be brief.
        layers.library's LockLayer is identical to LockLayerRom.

   INPUTS
        layer - pointer to Layer structure

   RESULTS
        The layer is locked and the task can render assuming the
        ClipRects will not change out from underneath it until
        an UnlockLayerRom is called.

   SEE ALSO
        UnlockLayerRom() layers.library/LockLayer() graphics/clip.h
```

```
graphics.library/MakeVPort                          graphics.library/MakeVPort

   NAME
        MakeVPort -- generate display copper list for a viewport.

   SYNOPSIS
        MakeVPort( view, viewport )
                   a0    a1

        void MakeVPort( struct View *, struct ViewPort * );

   FUNCTION
        Uses information in the View, ViewPort, ViewPort->RasInfo to
        construct and intermediate copper list for this ViewPort.

   INPUTS
        view - pointer to a View structure
        viewport - pointer to a ViewPort structure
                 The viewport must have valid pointer to a RasInfo.

   RESULTS
        constructs intermediate copper list and puts pointers in
        viewport.DspIns
        If the ColorMap ptr in ViewPort is NULL then it uses colors
        from the default color table.
        If DUALPF in Modes then there must be a second RasInfo pointed
        to by the first RasInfo

   BUGS

   SEE ALSO
        InitVPort() MrgCop() graphics/view.h intuition.library/MakeScreen()
        intuition.library/RemakeDisplay() intuition.library/RethinkDisplay()
```



graphics.library Page 83

```
graphics.library/ModeNotAvailable           graphics.library/ModeNotAvailable

   NAME
       ModeNotAvailable -- check to see if a DisplayID isn't available. (V36)

   SYNOPSIS
       error =  ModeNotAvailable( modeID )
       d0                         d0

       ULONG   ModeNotAvailable( ULONG);

   FUNCTION
       returns an error code, indicating why this modeID is not available,
       or NULL if there is no reason known why this mode should not be there.

   INPUTS
       modeID -- a 32 bit DisplayInfoRecord identifier.

   RESULT
       error -- a general indication of why this modeID is not available,
                or NULL if there is no reason why it shouldn't be available.

   NOTE
       ULONG return values from this function are a proper superset of the
       DisplayInfo.NotAvailable field (defined in graphics/displayinfo.h).

   BUGS

   SEE ALSO
       graphics/displayinfo.h, GetVPModeID()
```



graphics.library Page 84

```
graphics.library/Move                                   graphics.library/Move

   NAME
       Move -- Move graphics pen position.

   SYNOPSIS
       Move( rp,   x,    y)
             a1  d0:16 d1:16

       void Move( struct RastPort *, SHORT, SHORT );

   FUNCTION
       Move graphics pen position to (x,y) relative to upper left (0,0)
       of RastPort. This sets the starting point for subsequent Draw()
       and Text() calls.

   INPUTS
       rp - pointer to a RastPort structure
       x,y - point in the RastPort

   RESULTS

   BUGS

   SEE ALSO
       Draw graphics/rastport.h
```

```
graphics.library/MoveSprite                        graphics.library/MoveSprite

   NAME
        MoveSprite -- Move sprite to a point relative to top of viewport.

   SYNOPSIS
        MoveSprite(vp, sprite, x, y)
                   A0  A1      D0 D1

        void MoveSprite(struct ViewPort *,struct SimpleSprite *, WORD, WORD);

   FUNCTION
        Move sprite image to new place on display.

   INPUTS
        vp - pointer to ViewPort structure
             if vp = 0, sprite is positioned relative to View.
        sprite - pointer to SimpleSprite structure
        (x,y)  - new position relative to top of viewport or view.

   RESULTS
        Calculate the hardware information for the sprite and
        place it in the posctldata array. During next video display
        the sprite will appear in new position.

   BUGS
        Sprites really appear one pixel to the left of the position you specify.
        This bug affects the apparent display position of the sprite on the
        screen, but does not affect the numeric position relative to the
        viewport or view.

   SEE ALSO
        FreeSprite()  ChangeSprite()  GetSprite()  graphics/sprite.h
```

```
graphics.library/MrgCop                                graphics.library/MrgCop

   NAME
       MrgCop -- Merge together coprocessor instructions.

   SYNOPSIS
       MrgCop( View )
               A1

        void MrgCop( struct View * );

   FUNCTION
       Merge together the display, color, sprite and user coprocessor
       instructions into a single coprocessor instruction stream.  This
       essentially creates a per-display-frame program for the coprocessor.
       This function MrgCop is used, for example, by the graphics animation
       routines which effectively add information into an essentially
       static background display.  This changes some of the user
       or sprite instructions, but not those which have formed the
       basic display in the first place.  When all forms of coprocessor
       instructions are merged together, you will have a complete per-
       frame instruction list for the coprocessor.

       Restrictions:  Each of the coprocessor instruction lists MUST be
       internally sorted in min to max Y-X order.  The merge routines
       depend on this! Each list must be terminated using CEND(copperlist).

   INPUTS
       View - a pointer to the view structure whose coprocessor
              instructions are to be merged.

   RESULT

       The view structure will now contain a complete, sorted/merged
       list of instructions for the coprocessor, ready to be used by
       the display processor.  The display processor is told to use
       this new instruction stream through the instruction LoadView().

   BUGS

   SEE ALSO
       InitVPort() MakeVPort() LoadView() graphics/view.h
        intuition.library/RethinkDisplay()
```

graphics.library Page 87

```
graphics.library/NewRegion                        graphics.library/NewRegion

   NAME
       NewRegion -- Get an empty region.

   SYNOPSIS
       region = NewRegion()
        d0

        struct Region *NewRegion();

   FUNCTION
        Create a Region structure, initialize it to empty, and return
        a pointer it.

   RESULTS
        region - pointer to initialized region. If it could not allocate
                required memory region = NULL.

   INPUTS
        none

   BUGS

   SEE ALSO
        graphics/regions.h
```

graphics.library Page 88

```
graphics.library/NextDisplayInfo            graphics.library/NextDisplayInfo

   NAME
       NextDisplayInfo -- iterate current displayinfo identifiers (V36)

   SYNOPSIS
       next_ID = NextDisplayInfo(last_ID)
       D0                        D0

       ULONG NextDisplayInfo(ULONG);

   FUNCTION
       The basic iteration function with which to find all records in the
       graphics database.  Using each ID in succession, you can then call
       FindDisplayInfo() to obtain the handle associated with each ID.
       Each ID is a 32-bit Key which uniquely identifies one record.
       The INVALID_ID is special, and indicates the end-of-list.

   INPUTS
       last_ID - previous displayinfo identifier
                 or INVALID_ID if beginning iteration.

   RESULT
       next_ID - subsequent displayinfo identifier
                 or INVALID_ID if no more records.

   BUGS

   SEE ALSO
       FindDisplayInfo(), GetDisplayInfoData()
       graphics/displayinfo.h
```

```
graphics.library/OpenFont                              graphics.library/OpenFont

   NAME
       OpenFont -- Get a pointer to a system font.

   SYNOPSIS
       font = OpenFont(textAttr)
       D0              A0

       struct TextFont *OpenFont(struct TextAttr *);

   FUNCTION
       This function searches the system font space for the graphics
       text font that best matches the attributes specified.  The
       pointer to the font returned can be used in subsequent
       SetFont and CloseFont calls.  It is important to match this
       call with a corresponding CloseFont call for effective
       management of ram fonts.

   INPUTS
       textAttr - a TextAttr or XTextAttr structure that describes the
                  text font attributes desired.

   RESULT
       font is zero if the desired font cannot be found.  If the named
       font is found, but the size and style specified are not
       available, a font with the nearest attributes is returned.

   SEE ALSO
       CloseFont()  SetFont()
       diskfont.library/OpenDiskFont  graphics/text.h
```

```
graphics.library/OpenMonitor                        graphics.library/OpenMonitor

   NAME
      OpenMonitor -- open a named MonitorSpec (V36)

   SYNOPSIS
      mspc = OpenMonitor( monitor_name , display_id)
      d0                  a1             d0

      struct MonitorSpec *OpenMonitor( char *, ULONG );

   FUNCTION
      Locate and open a named MonitorSpec.

   INPUTS
      monitor_name - a pointer to a null terminated string.
      display_id - an optional 32 bit monitor/mode identifier

   RESULTS
      mspc - a pointer to an open MonitorSpec structure.
             NULL if MonitorSpec could not  be opened.

   NOTE
       if monitor_name is non-NULL, the monitor will be opened by name.
       if monitor_name is NULL the monitor will be opened by optional ID.
       if both monitor_name and display_id are NULL returns default monitor.

   BUGS

   SEE ALSO
      CloseMonitor() graphics/monitor.h
```

graphics.library Page 91

```
graphics.library/OrRectRegion                    graphics.library/OrRectRegion

   NAME
       OrRectRegion -- Perform 2d OR operation of rectangle
                       with region, leaving result in region.

   SYNOPSIS
       status = OrRectRegion(region,rectangle)
         d0                    a0      a1

        BOOL OrRectRegion( struct Region *, struct Rectangle * );

   FUNCTION
       If any portion of rectangle is not in the region then add
       that portion to the region.

   INPUTS
       region - pointer to Region structure
       rectangle - pointer to Rectangle structure

   RESULTS
        status - return TRUE if successful operation
                 return FALSE if ran out of memory

   BUGS

   SEE ALSO
        AndRectRegion() OrRegionRegion() graphics/regions.h
```

graphics.library Page 92

```
graphics.library/OrRegionRegion              graphics.library/OrRegionRegion

   NAME
       OrRegionRegion -- Perform 2d OR operation of one region
                       with second region, leaving result in second region

   SYNOPSIS
       status = OrRegionRegion(region1,region2)
         d0                        a0      a1

        BOOL OrRegionRegion( struct Region *, struct region * );

   FUNCTION
       If any portion of region1  is not in the region then add
       that portion to the region2

   INPUTS
       region1 - pointer to Region structure
       region2 - pointer to Region structure

   RESULTS
        status - return TRUE if successful operation
                 return FALSE if ran out of memory

   BUGS

   SEE ALSO
        OrRectRegion() graphics/regions.h
```

```
graphics.library/OwnBlitter                          graphics.library/OwnBlitter

   NAME
       OwnBlitter -- get the blitter for private usage

   SYNOPSIS
       OwnBlitter()

        void OwnBlitter( void );

   FUNCTION
        If blitter is available return immediately with the blitter
        locked for your exclusive use. If the blitter is not available
        put task to sleep. It will be awakened as soon as the blitter
        is available. When the task first owns the blitter the blitter
        may still be finishing up a blit for the previous owner. You
        must do a WaitBlit before actually using the blitter registers.

        Calls to OwnBlitter() do not nest. If a task that owns the
        blitter calls OwnBlitter() again, a lockup will result.
        (Same situation if the task calls a system function
        that tries to own the blitter).

   INPUTS
        NONE

   RETURNS
        NONE

   SEE ALSO
        DisownBlitter() WaitBlit()
```

```
graphics.library/PolyDraw                                  graphics.library/PolyDraw

   NAME
        PolyDraw -- Draw lines from table of (x,y) values.

   SYNOPSIS
        PolyDraw( rp, count , array )
                  a1   d0      a0

        void PolyDraw( struct RastPort *, WORD, WORD * );

   FUNCTION
        starting with the first pair in the array, draw connected lines to
        it and every successive pair.

   INPUTS
        rp - pointer to RastPort structure
        count -  number of (x,y) pairs in the array
        array - pointer to first (x,y) pair

   BUGS

   SEE ALSO
        Draw() Move() graphics/rastport.h
```

```
graphics.library/QBlit                                   graphics.library/QBlit

   NAME

        QBlit -- Queue up a request for blitter usage

   SYNOPSIS
        QBlit( bp )
               a1

        void QBlit( struct bltnode * );

   FUNCTION
        Link a request for the use of the blitter to the end of the
       current blitter queue.  The pointer bp points to a blit structure
       containing, among other things, the link information, and the
       address of your routine which is to be called when the blitter
       queue finally gets around to this specific request.  When your
       routine is called, you are in control of the blitter ... it is
       not busy with anyone else's requests.  This means that you can
       directly specify the register contents and start the blitter.
       See the description of the blit structure and the uses of QBlit
       in the section titled Graphics Support in the OS Kernel Manual.
        Your code must be written to run either in supervisor or user
        mode on the 68000.

   INPUTS
        bp - pointer to a blit structure

   RESULT
        Your routine is called when the blitter is ready for you.
        In general requests for blitter usage through this channel are
        put in front of those who use the blitter via OwnBlitter and
        DisownBlitter. However for small blits there is more overhead
        using the queuer than Own/Disown Blitter.

   BUGS

   SEE ALSO
        QBSBlit() hardware/blit.h
```

```
graphics.library/QBSBlit                                 graphics.library/QBSBlit

   NAME

        QBSBlit -- Synchronize the blitter request with the video beam.

   SYNOPSIS

        QBSBlit( bsp )
                 a1

        void QBSBlit( struct bltnode * );

   FUNCTION
        Call a user routine for use of the blitter, enqueued separately from
       the QBlit queue.  Calls the user routine contained in the blit
       structure when the video beam is located at a specified position
       onscreen.   Useful when you are trying to blit into a visible part
       of the screen and wish to perform the data move while the beam is
       not trying to display that same area.  (prevents showing part of
       an old display and part of a new display simultaneously).  Blitter
       requests on the QBSBlit queue take precedence over those on the
       regular blitter queue. The beam position is specified the blitnode.

   INPUTS
        bsp - pointer to a blit structure.  See description in the
             Graphics Support section of the manual for more info.

   RESULT
       User routine is called when the QBSBlit queue reaches this
       request AND the video beam is in the specified position.
        If there are lots of blits going on and the video beam
        has wrapped around back to the top it will call all the
        remaining bltnodes as fast as it can to try and catch up.

   BUGS
        Not very smart when getting blits from different tasks.
        They all get put in same queue so there are unfortunately
        some interdependencies with the beam syncing.

   SEE ALSO
        QBlit() hardware/blit.h
```

```
graphics.library/ReadPixel                          graphics.library/ReadPixel

   NAME
       ReadPixel -- read the pen number value of the pixel at a
                    specified x,y location within a certain RastPort.

   SYNOPSIS
       penno = ReadPixel( rp,    x,    y )
         d0               a1  d0:16 d1:16

        LONG ReadPixel( struct RastPort *, SHORT, SHORT );

   FUNCTION
       Combine the bits from each of the bit-planes used to describe
       a particular RastPort into the pen number selector which that
       bit combination normally forms for the system hardware selection
       of pixel color.

   INPUTS
       rp -  pointer to a RastPort structure
       (x,y) a point in the RastPort

   RESULT
       penno - the pen number of the pixel at (x,y) is returned.
               -1 is returned if the pixel cannot be read for some reason.

   BUGS

   SEE ALSO
       WritePixel()     graphics/rastport.h
```

```
graphics.library/ReadPixelArray8              graphics.library/ReadPixelArray8

   NAME
       ReadPixelArray8 -- read the pen number value of a rectangular array
       of pixels starting at a specified x,y location and continuing
       through to another x,y location within a certain RastPort. (V36)

   SYNOPSIS
       count = ReadPixelArray8(rp,xstart,ystart,xstop,ystop,array,temprp)
       D0                      A0 D0:16  D1:16  D2:16 D3:16 A2    A1

    LONG ReadPixelArray8(struct  RastPort *, UWORD, UWORD, UWORD, UWORD,
         UBYTE *, struct RastPort *);

   FUNCTION
       For each pixel in a rectangular region, combine the bits from each
       of the bit-planes used to describe a particular RastPort into the pen
       number selector which that bit combination normally forms for the
       system hardware selection of pixel color.

   INPUTS
       rp    -  pointer to a RastPort structure
       (xstart,ystart) - starting point in the RastPort
       (xstop,ystop)   - stopping point in the RastPort
       array - pointer to an array of ubytes from which to fetch the pixel data
               allocate at least ((((width+15)>>4)<<4)*(ystop-ystart+1)) bytes.
       temprp - temporary rastport (copy of rp with Layer set == NULL,
                temporary memory allocated for
                temprp->BitMap with Rows set == 1,
                temprp->BytesPerRow == (((width+15)>>4)<<1),
                and temporary memory allocated for
                temprp->BitMap->Planes[])

   RESULT
       For each pixel in the array:
           Pen - (0..255) number at that position is returned
       count   - the number of pixels read.

   NOTE
       xstop must be >= xstart
       ystop must be >= ystart

   BUGS

   SEE ALSO
       ReadPixel()  ReadPixelLine8()  graphics/rastport.h
```

```
graphics.library/ReadPixelLine8                    graphics.library/ReadPixelLine8

   NAME
       ReadPixelLine8 -- read the pen number value of a horizontal line
       of pixels starting at a specified x,y location and continuing
       right for count pixels. (V36)

   SYNOPSIS
       count = ReadPixelLine8(rp,xstart,ystart,width,array,temprp)
       D0                     A0 D0:16  D1:16  D2    A2    A1

       LONG ReadPixelLine8(struct RastPort *, UWORD, UWORD, UWORD,
            UBYTE *, struct RastPort * );

   FUNCTION
       For each pixel in a rectangular region, combine the bits from each
       of the bit-planes used to describe a particular RastPort into the pen
       number selector which that bit combination normally forms for the
       system hardware selection of pixel color.

   INPUTS
       rp     - pointer to a RastPort structure
       (x,y)  - a point in the RastPort
       width  - count of horizontal pixels to read
       array  - pointer to an array of UBYTEs from which to fetch the pixel data
                allocate at least (((width+15)>>4)<<4) bytes.
       temprp - temporary rastport (copy of rp with Layer set == NULL,
                temporary memory allocated for
                temprp->BitMap with Rows set == 1,
                temprp->BytesPerRow == (((width+15)>>4)<<1),
                and temporary memory allocated for
                temprp->BitMap->Planes[])

   RESULT
       For each pixel in the array:
           Pen - (0..255) number at that position is returned
       count   - the number of pixels read.

   NOTE
       width must be non negative

   BUGS

   SEE ALSO
       ReadPixel()  graphics/rastport.h
```

```
graphics.library/RectFill                                 graphics.library/RectFill

   NAME
      RectFill -- Fill a rectangular region in a RastPort.

   SYNOPSIS

       RectFill( rp, xmin, ymin, xmax, ymax)
                 a1  d0:16 d1:16 d2:16 d3:16

       void RectFill( struct RastPort *, SHORT, SHORT, SHORT, SHORT );

   FUNCTION
       Fills  the  rectangular  region  specified  by  the
       parameters  with the chosen pen  colors,  areafill
       pattern, and drawing mode. If no areafill pattern is
       specified, fill the rectangular region with the FgPen
       color, taking into account the drawing mode.

   INPUTS
       rp - pointer to a RastPort structure
       (xmin,ymin) (xmax,ymax) are the coordinates of the upper
               left corner and the lower right corner, respectively, of the
               rectangle.
   NOTE

       The following relation MUST be true:
               (xmax >= xmin) and (ymax >= ymin)

   BUGS
       Complement mode with FgPen complements all bitplanes.

   SEE ALSO
       AreaEnd() graphics/rastport.h
```

```
graphics.library/RemBob                                   graphics.library/RemBob

   NAME
        RemBob -- Macro to remove a Bob from the gel list.

   SYNOPSIS
        RemBob(bob)

        RemBob(struct Bob *);

   FUNCTION
        Marks a Bob as no-longer-required.  The gels internal code then
        removes the Bob from the list of active gels the next time
        DrawGList is executed. This is implemented as a macro.
        If the user is double-buffering the Bob, it could take two
        calls to DrawGList before the Bob actually disappears from
        the RastPort.

   INPUTS
        Bob = pointer to the Bob to be removed

   RESULT

   BUGS

   SEE ALSO
        RemIBob()  DrawGList()  graphics/gels.h  graphics/gfxmacros.h
```

```
graphics.library/RemFont                                  graphics.library/RemFont

   NAME
        RemFont -- Remove a font from the system list.

   SYNOPSIS
        RemFont(textFont)
                A1

        void RemFont(struct TextFont *);

   FUNCTION
        This function removes a font from the system, ensuring that
        access to it is restricted to those applications that
        currently have an active pointer to it: i.e. no new SetFont
        requests to this font are satisfied.

   INPUTS
        textFont - the TextFont structure to remove.

   RESULT

   BUGS

   SEE ALSO
        SetFont()  AddFont()  graphics/text.h
```

```
graphics.library/RemIBob                                   graphics.library/RemIBob

   NAME
       RemIBob -- Immediately remove a Bob from the gel list and the RastPort.

   SYNOPSIS
       RemIBob(bob, rp, vp)
               A0   A1  A2

       void RemIBob(struct Bob *, struct RastPort *, struct ViewPort *);

   FUNCTION
       Removes a Bob immediately by uncoupling it from the gel list and
       erases it from the RastPort.

   INPUTS
       bob = pointer to the Bob to be removed
       rp  = pointer to the RastPort if the Bob is to be erased
       vp  = pointer to the ViewPort for beam-synchronizing

   RESULT

   BUGS

   SEE ALSO
       InitGels()  RemVSprite()  graphics/gels.h
```

```
graphics.library/RemVSprite                          graphics.library/RemVSprite

   NAME
       RemVSprite -- Remove a VSprite from the current gel list.

   SYNOPSIS
       RemVSprite(vs)
                  A0

       void RemVSprite(struct VSprite *);

   FUNCTION
       Unlinks the VSprite from the current gel list.

   INPUTS
       vs = pointer to the VSprite structure to be removed from the gel list

   RESULT

   BUGS

   SEE ALSO
       InitGels()  RemIBob()  graphics/gels.h
```

```
graphics.library/ScalerDiv                             graphics.library/ScalerDiv

   NAME
       ScalerDiv -- Get the scaling result that BitMapScale would. (V36)

   SYNOPSIS
       result = ScalerDiv(factor, numerator, denominator)
       D0                 D0      D1         D2

       UWORD ScalerDiv(UWORD, UWORD, UWORD);

   FUNCTION
       Calculate the expression (factor*numerator/denominator) such
       that the result is the same as the width of the destination
       result of BitMapScale when the factor here is the width of
       the source, and the numerator and denominator are the
       XDestFactor and XSrcFactor for BitMapScale.

   INPUTS
       factor                 - a number in the range 0..16383
       numerator, denominator - numbers in the range 1..16383

   RESULT
       this returns factor*numerator/denominator
```

```
graphics.library/ScrollRaster                       graphics.library/ScrollRaster

   NAME
       ScrollRaster -- Push bits in rectangle in raster around by
                       dx,dy towards 0,0 inside rectangle.
   SYNOPSIS
       ScrollRaster(rp, dx, dy, xmin, ymin, xmax, ymax)
                    A1  D0  D1  D2    D3    D4    D5

       void ScrollRaster
            (struct RastPort *, WORD, WORD, WORD, WORD, WORD, WORD);

   FUNCTION
       Move the bits in the raster by (dx,dy) towards (0,0)
       The space vacated is RectFilled with BGPen.
       Limit the scroll operation to the rectangle defined
       by (xmin,ymin)(xmax,ymax). Bits outside will not be
       affected. If xmax,ymax is outside the rastport then use
       the lower right corner of the rastport.
       If you are dealing with a SimpleRefresh layered RastPort you
       should check rp->Layer->Flags & LAYER_REFRESH to see if
       there is any damage in the damage list.  If there is you should
       call the appropriate BeginRefresh(Intuition) or BeginUpdate(graphics)
       routine sequence.

   INPUTS
       rp - pointer to a RastPort structure
       dx,dy are integers that may be postive, zero, or negative
       xmin,ymin - upper left of bounding rectangle
       xmax,ymax - lower right of bounding rectangle

   EXAMPLE
       ScrollRaster(rp,0,1)    /* shift raster up by one row */
       ScrollRaster(rp,-1,-1)  /* shift raster down and to the right by 1 pixel

   BUGS
       In 1.2/V1.3 if you ScrollRaster a SUPERBITMAP exactly left or
       right, and there is no TmpRas attached to the RastPort, the system
       will allocate one for you, but will never free it or record its
       location. This bug has been fixed for V1.4.  The workaround for
       1.2/1.3 is to attach a valid TmpRas of size at least
       MAXBYTESPERROW to the RastPort before the call.

       Begining with V1.4 ScrollRaster adds the shifted areas into the
       damage list for SIMPLE_REFRESH windows. Due to unacceptable
       system overhead, the decision was made NOT to propagate this
       shifted area damage for SMART_REFRESH windows.

   SEE ALSO
       graphics/rastport.h
```

```
graphics.library/ScrollVPort                        graphics.library/ScrollVPort

   NAME
        ScrollVPort -- Reinterpret RasInfo information in ViewPort to reflect
                        the current Offset values.

   SYNOPSIS
        ScrollVPort( vp )
                     a0

        struct ViewPort *vp;

   FUNCTION
        After the programmer has adjusted the Offset values in
        the RasInfo structures of ViewPort, change the
        the copper lists to reflect the the Scroll positions.
        Changing the BitMap ptr in RasInfo and not changing the
        the Offsets will effect a double buffering affect.

   INPUTS
       vp - pointer to a ViewPort structure
             that is currently be displayed.
   RESULTS
        modifies hardware and intermediate copperlists to reflect
        new RasInfo

   BUGS
       pokes not fast enough to avoid some visible hashing of display

   SEE ALSO
        MakeVPort() MrgCop() LoadView()  graphics/view.h
```

```
graphics.library/SetAPen                                  graphics.library/SetAPen

   NAME
        SetAPen -- Set the primary pen for a RastPort.

   SYNOPSIS
        SetAPen( rp, pen )
                 a1  d0

        void SetAPen( struct RastPort *, UBYTE );

   FUNCTION
        Set the primary drawing pen for lines, fills, and text.

   INPUTS
        rp - pointer to RastPort structure.
        pen - (0-255)

   RESULT
        Changes the minterms in the RastPort to reflect new primary pen.
        Sets line drawer to restart pattern.

   BUGS

   SEE ALSO
        SetBPen() graphics/rastport.h
```

```
graphics.library/SetBPen                                graphics.library/SetBPen

   NAME
        SetBPen -- Set secondary pen for a RastPort

   SYNOPSIS
        SetBPen( rp, pen )
                 a1  d0

        void SetBPen( struct RastPort *, UBYTE );

   FUNCTION
        Set the secondary drawing pen for lines, fills, and text.

   INPUTS
        rp - pointer to RastPort structure.
        pen - (0-255)

   RESULT
        Changes the minterms in the RastPort to reflect new secondary pen.
        Sets line drawer to restart pattern.

   BUGS

    SEE ALSO
        SetAPen() graphics/rastport.h
```

```
graphics.library/SetCollision                      graphics.library/SetCollision

   NAME
        SetCollision -- Set a pointer to a user collision routine.

   SYNOPSIS
        SetCollision(num, routine, GInfo)
                     D0   A0       A1

        void SetCollision(ULONG, VOID (*)(), struct GelsInfo *);

   FUNCTION
        Sets a specified entry (num) in the user's collision vectors table
        equal to the address of the specified collision routine.

   INPUTS
        num     = collision vector number
        routine = pointer to the user's collision routine
        GInfo   = pointer to a GelsInfo structure

   RESULT

   BUGS

   SEE ALSO
        InitGels()  graphics/gels.h  graphics/rastport.h
```

```
graphics.library/SetDrMd                                   graphics.library/SetDrMd

   NAME
        SetDrMd -- Set drawing mode for a RastPort

   SYNOPSIS
        SetDrMd( rp, mode )
                 a1  d0:8

        void SetDrMd( struct RastPort *, UBYTE );

   FUNCTION
        Set the drawing mode for lines, fills and text.
        Get the bit definitions from rastport.h

   INPUTS
        rp - pointer to RastPort structure.
        mode - 0-255, some combinations may not make much sense.

   RESULT
        The mode set is dependant on the bits selected.
        Changes minterms to reflect new drawing mode.
        Sets line drawer to restart pattern.

   BUGS

   SEE ALSO
        SetAPen() SetBPen() graphics/rastport.h
```

```
graphics.library/SetFont                                   graphics.library/SetFont

   NAME
        SetFont -- Set the text font and attributes in a RastPort.

   SYNOPSIS
        SetFont(rp, font)
                A1   A0

        void SetFont(struct RastPort *, struct TextFont *);

   FUNCTION
        This function sets the font in the RastPort to that described
        by font, and updates the text attributes to reflect that
        change.  This function clears the effect of any previous
        soft styles.

   INPUTS
        rp   - the RastPort in which the text attributes are to be changed
        font - pointer to a TextFont structure returned from OpenFont()
               or OpenDiskFont()

   RESULT

   NOTES
        This function had previously been documented that it would
        accept a null font.  This practice is discouraged.
        o   Use of a RastPort with a null font with text routines has
            always been incorrect and risked the guru.
        o   Keeping an obsolete font pointer in the RastPort is no more
            dangerous than keeping a zero one there.
        o   SetFont(rp, 0) causes spurious low memory accesses under
            some system software releases.

        As of V36, the following Amiga font variants are no longer
        directly supported:
            fonts with NULL tf_CharSpace and non-NULL tf_CharKern.
            fonts with non-NULL tf_CharSpace and NULL tf_CharKern.
            fonts with NULL tf_CharSpace and NULL tf_CharKern with
                a tf_CharLoc size component greater than tf_XSize.
        Attempts to SetFont these one of these font variants will
        cause the system to modify your font to make it acceptable.

   BUGS
        Calling SetFont() on in-code TextFonts (ie fonts not
        OpenFont()ed) will result in a loss of 24 bytes from
        the system as of V36.
        This can be resolved by calling StripFont().

   SEE ALSO
        OpenFont()  StripFont()
        diskfont.library/OpenDiskFont()  graphics/text.h
```

graphics.library Page 113

```
graphics.library/SetOPen                                graphics.library/SetOPen

    NAME
        SetOPen -- Change the Area OutLine pen and turn on Outline
                        mode for areafills.

    SYNOPSIS
        SetOPen(rp, pen)

        void SetOPen( struct RastPort *, UBYTE );

    FUNCTION
        This is implemented as a c-macro.
        Pen is the pen number that will be used to draw a border
        around an areafill during AreaEnd().

    INPUTS
        rp = pointer to RastPort structure
        pen = number  between 0-255

    BUGS

    SEE ALSO
        AreaEnd() graphics/gfxmacros.h graphics/rastport.h
```

graphics.library Page 114

```
graphics.library/SetRast                                graphics.library/SetRast

    NAME
        SetRast - Set an entire drawing area to a specified color.

    SYNOPSIS
        SetRast( rp, pen )
                 a1  d0

         void SetRast( struct RastPort *, UBYTE );

    FUNCTION
        Set the entire contents of the specified RastPort to the
        specified pen.

    INPUTS
        rp - pointer to RastPort structure
        pen - the pen number (0-255) to jam into bitmap

    RESULT
        All pixels within the drawing area are set to the
         selected pen number.

    BUGS

    SEE ALSO
         RectFill() graphics/rastport.h
```

```
graphics.library/SetRGB4                                graphics.library/SetRGB4

   NAME
      SetRGB4 -- Set one color register for this viewport.

   SYNOPSIS
      SetRGB4(  vp, n,   r,    g,    b)
                a0  d0  d1:4  d2:4  d3:4

       void SetRGB4( struct ViewPort *, SHORT, UBYTE, UBYTE, UBYTE );

   FUNCTION
       Change the color look up table so that this viewport displays
       the color (r,g,b) for pen number n.

   INPUTS
       vp - pointer to  viewport structure
      n - the color number (range from 0 to 31)
      r - red level (0-15)
      g - green level (0-15)
      b - blue level (0-15)

   RESULT
       If there is a ColorMap for this viewport, then the value will
       be stored in the ColorMap.
      The selected color register is changed to match your specs.
       If the color value is unused then nothing will happen.

   BUGS

   SEE ALSO
      LoadRGB4() GetRGB4() graphics/view.h
```

```
graphics.library/SetRGB4CM                            graphics.library/SetRGB4CM

   NAME
       SetRGB4CM -- Set one color register for this ColorMap.

   SYNOPSIS
       SetRGB4CM(  cm,  n,   r,    g,    b)
                   a0  d0  d1:4  d2:4  d3:4

       void SetRGB4CM( struct ColorMap *, SHORT, UBYTE, UBYTE, UBYTE );

   INPUTS
        cm = colormap
       n = the number of the color register to set. Ranges from 0 to 31
           on current amiga displays.
       r = red level (0-15)
       g = green level (0-15)
       b = blue level (0-15)

   RESULT
        Store the (r,g,b) triplet at index n of the ColorMap structure.
       This function can be used to set up a ColorMap before before
        linking it into a viewport.

   BUGS

   SEE ALSO
       GetColorMap() GetRGB4() SetRGB4() graphics/view.h
```

```
graphics.library/SetSoftStyle                     graphics.library/SetSoftStyle

   NAME
        SetSoftStyle -- Set the soft style of the current font.

   SYNOPSIS
        newStyle = SetSoftStyle(rp, style, enable)
        D0                      A1  D0     D1

        ULONG SetSoftStyle(struct RastPort *, ULONG, ULONG);

   FUNCTION
        This function alters the soft style of the current font.  Only
        those bits that are also set in enable are affected.  The
        resulting style is returned, since some style request changes
        will not be honored when the implicit style of the font
        precludes changing them.

   INPUTS
        rp     - the RastPort from which the font and style
                 are extracted.
        style  - the new font style to set, subject to enable.
        enable - those bits in style to be changed.  Any set bits here
                 that would not be set as a result of AskSoftStyle will
                 be ignored, and the newStyle result will not be as
                 expected.

   RESULTS
        newStyle - the resulting style, both as a result of previous
                   soft style selection, the effect of this function,
                   and the style inherent in the set font.

   BUGS

   SEE ALSO
        AskSoftStyle()  graphics/text.h
```

```
graphics.library/SortGList                               graphics.library/SortGList

   NAME
        SortGList -- Sort the current gel list, ordering its y,x coordinates.

   SYNOPSIS
        SortGList(rp)
                  A1

        void SortGList(struct RastPort *);

   FUNCTION
        Sorts the current gel list according to the gels' y,x coordinates.
        This sorting is essential before calls to DrawGList or DoCollision.

   INPUTS
        rp = pointer to the RastPort structure containing the GelsInfo

   RESULT

   BUGS

   SEE ALSO
        InitGels()  DoCollision()  DrawGList()  graphics/rastport.h
```

```
graphics.library/StripFont                          graphics.library/StripFont

   NAME
       StripFont -- remove the tf_Extension from a font (V36)

   SYNOPSIS
       StripFont(font)
                 A0

       VOID StripFont(struct TextFont *);
```

```
graphics.library/SyncSBitMap                      graphics.library/SyncSBitMap

   NAME
       SyncSBitMap --   Syncronize Super BitMap with whatever is
                        in the standard Layer bounds.

   SYNOPSIS
       SyncSBitMap( layer )
                    a0

        void SyncSBitMap( struct Layer * );

   FUNCTION
       Copy all bits from ClipRects in Layer into Super BitMap
        BitMap.  This is used for those functions that do not
        want to deal with the ClipRect structures but do want
        to be able to work with a SuperBitMap Layer.

   INPUTS
        layer - pointer to a Layer that has a SuperBitMap
                The Layer should already be locked by the caller.

   RESULT
        After calling this function, the programmer can manipulate
        the bits in the superbitmap associated with the layer.
        Afterwards, the programmer should call CopySBitMap to
        copy the bits back into the onscreen layer.

   BUGS

   SEE ALSO
        CopySBitMap() graphics/clip.h
```

```
graphics.library/Text                                 graphics.library/Text

   NAME
       Text -- Write text characters (no formatting).

   SYNOPSIS
       Text(rp, string, length)
            A1  A0      D0-0:16

       void Text(struct RastPort *, STRPTR, WORD);

   FUNCTION
       This graphics function writes printable text characters to the
       specified RastPort at the current position.  No control meaning
       is applied to any of the characters, thus only text on the
       current line is output.

       The current position in the RastPort is updated to the next
       character position.
       If the characters displayed run past the RastPort boundary,
       the current position is truncated to the boundary, and
       thus does not equal the old position plus the text length.

   INPUTS
       rp     - a pointer to the RastPort which describes where the
                text is to be output
       string - the address of string to output
       length - the number of characters in the string.
                If zero, there are no characters to be output.

   NOTES
       o   This function may use the blitter.
       o   Changing the text direction with RastPort->TxSpacing is
           not supported.

   BUGS
       For V34 and earlier:
       o   The maximum string length (in pixels) is limited to
           (1024 - 16 = 1008) pixels wide.
       o   A text string whose last character(s) have a
           tf_CharLoc size component that extends to the right of
           the rightmost of the initial and final CP positions
           will be (inappropriately) clipped.

   SEE ALSO
       Move()  TextLength()  graphics/text.h  graphics/rastport.h
```

```
graphics.library/TextExtent                     graphics.library/TextExtent

   NAME
       TextExtent -- Determine raster extent of text data. (V36)

   SYNOPSIS
       TextExtent(rp, string, count, textExtent)
                  A1  A0      D0:16  A2

       void textExtent(struct RastPort *, STRPTR, WORD,
            struct TextExtent *);

   FUNCTION
       This function determines a more complete metric of the space
       that a text string would render into than the TextLength()
       function.

   INPUTS
       rp     - a pointer to the RastPort which describes where the
                text attributes reside.
       string - the address of the string to determine the length of.
       count  - the number of characters in the string.
               If zero, there are no characters in the string.
       textExtent - a structure to hold the result.

   RESULTS
       textExtent is filled in as follows:
           te_Width  - same as TextLength() result: the rp_cp_x
                       advance that rendering this text would cause.
           te_Height - same as tf_YSize.  The height of the
                       font.
           te_Extent.MinX - the offset to the left side of the
                       rectangle this would render into.  Often zero.
           te_Extent.MinY - same as -tf_Baseline.  The offset
                       from the baseline to the top of the rectangle
                       this would render into.
           te_Extent.MaxX - the offset of the left side of the
                       rectangle this would render into.  Often the
                       same as te_Width-1.
           te_Extent.MaxY - same as tf_YSize-tf_Baseline-1.
                       The offset from the baseline to the bottom of
                       the rectanangle this would render into.

   SEE ALSO
       TextLength()  Text()  TextFit()
       graphics/text.h  graphics/rastport.h
```

graphics.library

Page 123

```
graphics.library/TextFit                                graphics.library/TextFit

   NAME
       TextFit - count characters that will fit in a given extent (V36)

   SYNOPSIS
       chars = TextFit(rastport, string, strLen, textExtent,
       D0              A1        A0      D0      A2
               constrainingExtent, strDirection,
               A3                  D1
               constrainingBitWidth, constrainingBitHeight)
               D2                    D3

       ULONG TextFit(struct RastPort *, STRPTR, UWORD,
           struct TextExtent *, struct TextExtent *, WORD, UWORD, UWORD);

   FUNCTION
       This function determines how many of the characters of the
       provided string will fit into the space described by the
       constraining parameters.  It also returns the extent of
       that number of characters.

   INPUTS
       rp     - a pointer to the RastPort which describes where the
                text attributes reside.
       string - the address of string to determine the constraint of
       strLen - The number of characters in the string.
                If zero, there are no characters in the string.
       textExtent - a structure to hold the extent result.
       constrainingExtent - the extent that the text must fit in.
           This can be NULL, indicating only the constrainingBit
           dimensions will describe the constraint.
       strDirection - the offset to add to the string pointer to
           get to the next character in the string.  Usually 1.
           Set to -1 and the string to the end of the string to
           perform a TextFit() anchored at the end.  No other value
           is valid.
       constrainingBitWidth - an alternative way to specify the
           rendering box constraint width that is independent of
           the rendering origin.  Range 0..32767.
       constrainingBitHeight - an alternative way to specify the
           rendering box constraint height that is independent of
           the rendering origin.  Range 0..32767.

   RESULTS
       chars - the number of characters from the origin of the
               given string that will fit in both the constraining
               extent (which specifies a CP bound and a rendering
               box relative to the origin) and in the rendering width
               and height specified.

   NOTES
       The result is zero chars and an empty textExtent when the fit
       cannot be performed.  This occurs not only when no text will
       fit in the provided constraints, but also when:
       -   the RastPort rp's rp_TxSpacing sign and magnitude is so
           great it reverses the path of the text.
       -   the constrainingExtent does not include x = 0.

   SEE ALSO
       TextExtent()  TextLength()  Text()
       graphics/text.h  graphics/rastport.h
```

graphics.library

Page 124

```
graphics.library/TextLength                          graphics.library/TextLength

   NAME
       TextLength -- Determine raster length of text data.

   SYNOPSIS
       length = TextLength(rp, string, count)
       D0                  A1  A0      D0:16

       WORD TextLength(struct RastPort *, STRPTR, WORD);

   FUNCTION
       This graphics function determines the length that text data
       would occupy if output to the specified RastPort with the
       current attributes.  The length is specified as the number of
       raster dots: to determine what the current position would be
       after a Write() using this string, add the length to cp_x
       (cp_y is unchanged by Write()).  Use the newer TextExtent() to
       get more information.

   INPUTS
       rp     - a pointer to the RastPort which describes where the
                text attributes reside.
       string - the address of string to determine the length of
       count  - the string length.  If zero, there are no characters
                in the string.

   RESULTS
       length - the number of pixels in x this text would occupy, not
                including any negative kerning that may take place at
                the beginning of the text string, nor taking into
                account the effects of any clipping that may take
                place.

   NOTES
       Prior to V36, the result length occupied only the low word of
       d0 and was not sign extended into the high word.

   BUGS
       A length that would overflow single word arithmatic is not
       calculated correctly.

   SEE ALSO
       TextExtent()  Text()  TextFit()
       graphics/text.h  graphics/rastport.h
```

```
graphics.library/UnlockLayerRom                    graphics.library/UnlockLayerRom

   NAME
        UnlockLayerRom -- Unlock Layer structure by rom(gfx lib) code.

   SYNOPSIS
        UnlockLayerRom( layer )
                         a5

        void UnlockLayerRom( struct Layer * );

   FUNCTION
        Release the lock on this layer. If the same task has called
        LockLayerRom more than once than the same number of calls to
        UnlockLayerRom must happen before the layer is actually freed
        so that other tasks may use it.
        This call does destroy scratch registers.
        This call is identical to UnlockLayer (layers.library).

   INPUTS
        layer - pointer to Layer structure

   BUGS

   SEE ALSO
        LockLayerRom() layers.library/UnlockLayer() graphics/clip.h
```

```
graphics.library/VBeamPos                                    graphics.library/VBeamPos

   NAME
        VBeamPos -- Get vertical beam position at this instant.

   SYNOPSIS
        pos = VBeamPos()
         d0

        LONG VBeamPos( void );

   FUNCTION
        Get the vertical beam position from the hardware.

   INPUTS
        none

   RESULT
        interrogates hardware for beam position and returns value.
        valid results in are the range of 0-511.
        Because of multitasking, the actual value returned may have
        no use. If you are the highest priority task then the value
        returned should be close, within 1 line.

   BUGS

   SEE ALSO
```

```
graphics.library/VideoControl                    graphics.library/VideoControl

   NAME
       VideoControl -- Modify the operation of a ViewPort's ColorMap (V36)

   SYNOPSIS
       error = VideoControl( cm , tags )
       d0                    a0   a1

        ULONG VideoControl( struct ColorMap *, struct TagItem * );

   FUNCTION
        Process the commands in the VideoControl command TagItem buffer
        using cm as the target, with respect to its "attached" ViewPort.

        viewport commands:

        VTAG_ATTACH_CM     [_SET        | _GET] -- set\get attached viewport
        VTAG_VIEWPORTEXTRA [_SET        | _GET] -- set\get attached vp_extra
        VTAG_NORMAL_DISP   [_SET        | _GET] -- set\get DisplayInfoHandle
                                                    (natural mode)
        VTAG_COERCE_DISP   [_SET        | _GET] -- set\get DisplayInfoHandle
                                                    (coerced mode)

        genlock commands:

        VTAG_BORDERBLANK   [_SET | _CLR | _GET] -- on\off\inquire blanking
        VTAG_BORDERNOTRANS [_SET | _CLR | _GET] -- on\off\inquire notransparency
        VTAG_CHROMAKEY     [_SET | _CLR | _GET] -- on\off\inquire chroma mode
        VTAG_BITPLANEKEY   [_SET | _CLR | _GET] -- on\off\inquire bitplane mode
        VTAG_CHROMA_PEN    [_SET | _CLR | _GET] -- set\clr\get chromakey pen #
        VTAG_CHROMA_PLANE  [_SET |      | _GET] -- set\get bitplanekey plane #

        buffer commands:

        VTAG_NEXTBUF_CM                         -- link to further commands
        VTAG_END_CM                             -- terminate command buffer


   INPUTS
        cm   = pointer to struct ColorMap obtained via GetColorMap().
        tags = pointer to a table of videocontrol tagitems.

   RESULT
        error = NULL if no error occured in the control operation.
       (non-NULL if bad colormap pointer, no tagitems or bad tag)

        The operating characteristics of the ColorMap and its attached
        ViewPort are modified. The result will be incorporated into the
        ViewPort when its copper lists are reassembled via MakeVPort().

   BUGS

   SEE ALSO
        graphics/videocontrol.h, GetColorMap(), FreeColorMap()
```

```
graphics.library/WaitBlit                                 graphics.library/WaitBlit

   NAME
       WaitBlit -- Wait for the blitter to be finished before proceeding
                   with anything else.

   SYNOPSIS
       WaitBlit()

        void WaitBlit( void );

   FUNCTION
        WaitBlit returns when the blitter is idle. This function should
        normally only be used when dealing with the blitter in a
        synchronous manner, such as when using OwnBlitter and DisownBlitter.
        WaitBlit does not wait for all blits queued up using QBlit or
        QBSBlit. You should call WaitBlit if you are just about to modify or
        free some memory that the blitter may be using.

   INPUTS
        none

   RESULT
       Your program waits until the blitter is finished.
        This routine does not use any the CPU registers.
        do/d1/a0/a1 are preserved by this routine.
        It may change the condition codes though.

   BUGS
        When examining bits with the CPU right after a blit, or when freeeing
        temorary memory used by the blitter, a WaitBlit() may be required.

        Note that many graphics calls fire up the blitter, and let it run.
        The CPU does not need to wait for the blitter to finish before
        returning.

        Because of a bug in agnus (prior to all revisions of fat agnus)
        this code may return too soon when the blitter has, in fact, not
        started the blit yet, even though BltSize has been written.

        This most often occurs in a heavily loaded systen with extended memory,
        HIRES, and 4 bitplanes.

        WaitBlit currently tries to avoid this agnus problem by testing
        the BUSY bit multiple times to make sure the blitter has started.
        If the blitter is BUSY at first check, this function busy waits.

        This initial hardware bug was fixed as of the first "Fat Agnus" chip,
        as used in all A500 and A2000 computers.

        Because of a different bug in agnus (currently all revisions thru ECS)
        this code may return too soon when the blitter has, in fact, not
        stopped the blit yet, even though blitter busy has been cleared.

        This most often occurs in a heavily loaded systen with extended memory,
        in PRODUCTIVITY mode, and 2 bitplanes.

        WaitBlit currently tries to avoid this agnus problem by testing
        the BUSY bit multiple times to make sure the blitter has really
        written its final word of desination data.

   SEE ALSO
        OwnBlitter() DisownBlitter() hardware/blit.h
```

```
graphics.library/WaitBOVP                               graphics.library/WaitBOVP

   NAME
       WaitBOVP -- Wait till vertical beam reached bottom of
                   this viewport.

   SYNOPSIS
       WaitBOVP( vp )
                 a0

       void WaitBOVP( struct ViewPort * );

   FUNCTION
       Returns when the vertical beam has reached the bottom of this viewport

   INPUTS
       vp - pointer to ViewPort structure

   RESULT
       This function will return sometime after the beam gets beyond
       the bottom of the viewport.  Depending on the multitasking load
       of the system, the actual beam position may be different than
       what would be expected in a lightly loaded system.

   BUGS
       Horrors! This function currently busy waits waiting for the
       beam to get to the right place.  It should use the copper
       interrupt to trigger and send signals like WaitTOF does.

   SEE ALSO
       WaitTOF() VBeamPos()
```

```
graphics.library/WaitTOF                                  graphics.library/WaitTOF

   NAME
       WaitTOF -- Wait for the top of the next video frame.

   SYNOPSIS
       WaitTOF()

        void WaitTOF( void );

   FUNCTION
       Wait  for vertical blank to occur and all vertical blank
       interrupt routines to complete before returning to caller.

   INPUTS
       none

   RESULT
        Places this task on the TOF wait queue. When the vertical blank
        interupt comes around, the interrupt service routine will fire off
        signals to all the tasks doing WaitTOF. The highest priority task
        ready will get to run then.

   BUGS

   SEE ALSO
        exec.library/Wait() exec.library/Signal()
```

```
graphics.library/WeighTAMatch                    graphics.library/WeighTAMatch

   NAME
        WeighTAMatch -- Get a measure of how well two fonts match. (V36)

   SYNOPSIS
        weight = WeighTAMatch(reqTextAttr, targetTextAttr, targetTags)
        D0                    A0           A1              A2

        WORD WeighTAMatch(struct TTextAttr *, struct TextAttr *,
             struct TagItem *);

   FUNCTION
        This function provides a metric to describe how well two fonts
        match.  This metric ranges from MAXFONTMATCHWEIGHT (perfect match)
        through lower positive numbers to zero (unsuitable match).

   INPUTS
        reqTextAttr    - the text attributes requested.
        targetTextAttr - the text attributes of a potential match.
        targetTags     - tags describing the extended target attributes, or
                         zero if not available.

        The [t]ta_Name fields of the [T]TextAttr structures are not used.

        The tags affect the weight only when both a) the reqTextAttr
        has the FSF_TAGGED bit set in ta_Style, and b) targetTags is
        not zero.  To fairly compare two different weights, the inclusion
        or exclusion of tags in the weighing must be the same for both.

   RESULTS
        weight -- a positive weight describes suitable matches, in
              increasing desirability.  MAXFONTMATCHWEIGHT is a perfect
              match.  A zero weight is an unsuitable match.

   SEE ALSO
        OpenFont()
```

```
graphics.library/WritePixel                        graphics.library/WritePixel

   NAME
       WritePixel -- Change the pen num of one specific pixel in a
                     specified RastPort.

   SYNOPSIS
       error = WritePixel(  rp, x,  y)
         d0                 a1 D0  D1

        LONG WritePixel( struct RastPort *, SHORT, SHORT );

   FUNCTION
       Changes the pen number of the selected pixel in the specified
       RastPort to that currently specified by PenA, the primary
       drawing pen. Obeys minterms in RastPort.

   INPUTS
       rp - a pointer to the RastPort structure
       (x,y) - point within the RastPort at which the selected
          pixel is located.

   RESULT
       error = 0 if pixel succesfully changed
             = -1 if (x,y) is outside the RastPort

   BUGS

   SEE ALSO
       ReadPixel() graphics/rastport.h
```

```
graphics.library/WritePixelArray8          graphics.library/WritePixelArray8

   NAME
       WritePixelArray8 -- write the pen number value of a rectangular array
       of pixels starting at a specified x,y location and continuing
       through to another x,y location within a certain RastPort. (V36)

   SYNOPSIS
       count = WritePixelArray8(rp,xstart,ystart,xstop,ystop,array,temprp)
       D0                       A0 D0:16  D1:16  D2:16 D3:16  A2   A1

       LONG WritePixelArray8(struct  RastPort *, UWORD, UWORD,
            UWORD, UWORD, UBYTE *, struct  RastPort *);

   FUNCTION
       For each pixel in a rectangular region, decode the pen number selector
       from a linear array of pen numbers into the bit-planes used to describe
       a particular rastport.

   INPUTS
       rp     -  pointer to a RastPort structure
       (xstart,ystart) -  starting point in the RastPort
       (xstop,ystop)   -  stopping point in the RastPort
       array  - pointer to an array of UBYTEs from which to fetch the
                pixel data. Allocate at least
                ((((width+15)>>4)<<4)*(ystop-ystart+1)) bytes.
       temprp - temporary rastport (copy of rp with Layer set == NULL,
                temporary memory allocated for
                temprp->BitMap with Rows set == 1,
                temprp->BytesPerRow == (((width+15)>>4)<<1),
                and temporary memory allocated for
                temprp->BitMap->Planes[])

   RESULT
       For each pixel in the array:
           Pen - (0..255) number at that position is returned

   NOTE
       xstop must be >= xstart
       ystop must be >= ystart

   BUGS

   SEE ALSO
       WritePixel()  graphics/rastport.h
```

```
graphics.library/WritePixelLine8            graphics.library/WritePixelLine8

   NAME
       WritePixelLine8 -- write the pen number value of a horizontal line
       of pixels starting at a specified x,y location and continuing
       right for count pixels. (V36)

   SYNOPSIS
       count = WritePixelLine8(rp,xstart,ystart,width,array,temprp)
       D0                      A0 D0:16  D1:16  D2    A2    A1

       LONG WritePixelLine8(struct RastPort *, UWORD, UWORD,
            UWORD, UBYTE *, struct RastPort *);

   FUNCTION
       For each pixel in a horizontal region, decode the pen number selector
       from a linear array of pen numbers into the bit-planes used to describe
       a particular rastport.

   INPUTS
       rp    -  pointer to a RastPort structure
       (x,y) - a point in the RastPort
       width - count of horizontal pixels to write
       array - pointer to an array of UBYTEs from which to fetch the pixel data
               allocate at least (((width+15)>>4)<<4) bytes.
       temprp - temporary rastport (copy of rp with Layer set == NULL,
                temporary memory allocated for
                temprp->BitMap with Rows set == 1,
                temprp->BytesPerRow == (((width+15)>>4)<<1),
                and temporary memory allocated for
                temprp->BitMap->Planes[])

   RESULT
       For each pixel in the array:
           Pen - (0..255) number at that position is returned

   NOTE
       width must be non negative

   BUGS

   SEE ALSO
       WritePixel()  graphics/rastport.h
```

graphics.library Page 135

```
graphics.library/XorRectRegion                    graphics.library/XorRectRegion

   NAME
       XorRectRegion -- Perform 2d XOR operation of rectangle
                        with region, leaving result in region

   SYNOPSIS
       status = XorRectRegion(region,rectangle)
         d0                      a0      a1

        BOOL XorrectRegion( struct Region *, struct Rectangle * );

   FUNCTION
        Add portions of rectangle to region if they are not in
        the region.
        Remove portions of rectangle from region if they are
        in the region.

   INPUTS
       region - pointer to Region structure
       rectangle - pointer to Rectangle structure

   RESULTS
        status - return TRUE if successful operation
                 return FALSE if ran out of memory

   BUGS

   SEE ALSO
        OrRegionRegion() AndRegionRegion() graphics/regions.h
```

graphics.library Page 136

```
graphics.library/XorRegionRegion                graphics.library/XorRegionRegion

   NAME
       XorRegionRegion -- Perform 2d XOR operation of one region
                          with second region, leaving result in second region

   SYNOPSIS
       status = XorRegionRegion(region1,region2)
         d0                          a0      a1

        BOOL XorRegionRegion( struct Region *, struct Region * );

   FUNCTION
        Join the regions together. If any part of region1 overlaps
        region2 then remove that from the new region.

   INPUTS
       region1      = pointer to Region structure
       region2      = pointer to Region structure

   RESULTS
        status - return TRUE if successful operation
                 return FALSE if ran out of memory

   BUGS
```

TABLE OF CONTENTS

icon.library/AddFreeList
icon.library/BumpRevision
icon.library/DeleteDiskObject
icon.library/FindToolType
icon.library/FreeDiskObject
icon.library/FreeFreeList
icon.library/GetDefDiskObject
icon.library/GetDiskObject
icon.library/GetDiskObjectNew
icon.library/MatchToolValue
icon.library/PutDefDiskObject
icon.library/PutDiskObject

```
icon.library/AddFreeList                                 icon.library/AddFreeList

   NAME
       AddFreeList - add memory to a free list.

   SYNOPSIS
       status = AddFreeList(free, mem, len)
        D0                    A0    A1   A2

        BOOL AddFreeList(struct FreeList *, APTR, ULONG);

   FUNCTION
        This routine adds the specified memory to the free list.
        The free list will be extended (if required).  If there
        is not enough memory to complete the call, a null is returned.

        Note that AddFreeList does NOT allocate the requested memory.
        It only records the memory in the free list.

   INPUTS
        free -- a pointer to a FreeList structure
        mem -- the base of the memory to be recorded
        len -- the length of the memory to be recorded

   RESULTS
        status -- TRUE if the call succeeded else FALSE;

   SEE ALSO
        AllocEntry(), FreeEntry(), FreeFreeList()

   BUGS
        None
```

```
icon.library/BumpRevision                              icon.library/BumpRevision

   NAME
       BumpRevision - reformat a name for a second copy.

   SYNOPSIS
       result = BumpRevision(newbuf, oldname)
        D0                     A0      A1

        char *BumpRevision(char *, char *);

   FUNCTION
        BumpRevision takes a name and turns it into a "copy_of_name".
        It knows how to deal with copies of copies.  The routine
        will truncate the new name to the maximum dos name size
        (currently 30 characters).

   INPUTS
        newbuf - the new buffer that will receive the name
                 (it must be at least 31 characters long).
        oldname - the original name

   RESULTS
        result - a pointer to newbuf

   EXAMPLE
        oldname                            newbuf
        -------                            ------
        "foo"                              "copy_of_foo"
        "copy_of_foo"                      "copy_2_of_foo"
        "copy_2_of_foo"                    "copy_3_of_foo"
        "copy_199_of_foo"                  "copy_200_of_foo"
        "copy foo"                         "copy_of_copy foo"
        "copy_0_of_foo"                    "copy_1_of_foo"
        "012345678901234567890123456789"  "copy_of_0123456789012345678901"

   SEE ALSO

   BUGS
        None
```

```
icon.library/DeleteDiskObject                    icon.library/DeleteDiskObject

   NAME
       DeleteDiskObject - Delete a Workbench disk object from disk.     (V37)

   SYNOPSIS
       result = DeleteDiskObject(name)
        D0                        A0

        BOOL DeleteDiskObject(char *);

   FUNCTION
        This routine will try to delete a Workbench disk object from disk.
        The name parameter will have a ".info" postpended to it, and the
        info file of that name will be deleted.  If the call fails,
        it will return zero.  The reason for the failure may be obtained
        via IoErr().

        This call also updates the Workbench screen if needed.

        Using this routine protects you from any future changes to
        the way icons are stored within the system.

   INPUTS
        name -- name of the object (char *)

   RESULTS
        result -- TRUE if it worked, false if not.

   EXAMPLE

        error=NULL;

        *Check if you have the right library version*

        if (((struct Library *)IconBase)->lib_Version > 36)
        {
                if (!DeleteDiskObject(name)) error=IoErr();
        }
        else
        {
                * Delete name plus ".info" *
        }

        if (error)
        {
                * Do error routine...*
        }

   SEE ALSO
        PutDiskObject(), GetDiskObject(), FreeDiskObject()

   BUGS
        None
```

```
icon.library/FindToolType                              icon.library/FindToolType

   NAME
       FindToolType - find the value of a ToolType variable.

   SYNOPSIS
       value = FindToolType(toolTypeArray, typeName)
        D0                  A0            A1

        char *FindToolType(char **, char *);

   FUNCTION
        This function searches a tool type array for a given entry,
        and returns a pointer to that entry.  This is useful for
        finding standard tool type variables.  The returned
        value is not a new copy of the string but is only
        a pointer to the part of the string after typeName.

   INPUTS
        toolTypeArray - an array of strings (char **).
        typeName - the name of the tooltype entry (char *).

   RESULTS
        value - a pointer to a string that is the value bound to typeName,
                or NULL if typeName is not in the toolTypeArray.

   EXAMPLE
        Assume the tool type array has two strings in it:
            "FILETYPE=text"
            "TEMPDIR=:t"

        FindToolType( toolTypeArray, "FILETYPE" ) returns "text"
        FindToolType( toolTypeArray, "filetype" ) returns "text"
        FindToolType( toolTypeArray, "TEMPDIR" )  returns ":t"
        FindToolType( toolTypeArray, "MAXSIZE" )  returns NULL

   SEE ALSO
        MatchToolValue()

   BUGS
        None
```

```
icon.library/FreeDiskObject                                icon.library/FreeDiskObject

   NAME
       FreeDiskObject - free all memory in a Workbench disk object.

   SYNOPSIS
       FreeDiskObject(diskobj)
                      A0

        void FreeDiskObject(struct DiskObject *);

   FUNCTION
        This routine frees all memory in a Workbench disk object, and the
        object itself.  It is implemented via FreeFreeList().

        GetDiskObject() takes care of all the initialization required
        to set up the object's free list.  This procedure may ONLY
        be called on a DiskObject allocated via GetDiskObject().

   INPUTS
        diskobj -- a pointer to a DiskObject structure

   RESULTS
        None

   SEE ALSO
        GetDiskObject(), PutDiskObject(), DeleteDiskObject(), FreeFreeList()

   BUGS
        None
```

icon.library Page 7

```
icon.library/FreeFreeList                              icon.library/FreeFreeList

   NAME
       FreeFreeList - free all memory in a free list.

   SYNOPSIS
       FreeFreeList(free)
                    A0

        void FreeFreeList(struct FreeList *);

   FUNCTION
        This routine frees all memory in a free list, and the
        free list itself.  It is useful for easily getting
        rid of all memory in a series of structures.  There is
        a free list in a Workbench object, and this contains
        all the memory associated with that object.

        A FreeList is a list of MemList structures.  See the
        MemList and MemEntry documentation for more information.

        If the FreeList itself is in the free list, it must be
        in the first MemList in the FreeList.

   INPUTS
        free -- a pointer to a FreeList structure

   RESULTS
        None

   SEE ALSO
        AllocEntry(), FreeEntry(), AddFreeList()

   BUGS
        None
```

icon.library Page 8

```
icon.library/GetDefDiskObject                      icon.library/GetDefDiskObject

   NAME
        GetDefDiskObject - read default wb disk object from disk.        (V36)

   SYNOPSIS
        diskobj = GetDefDiskObject(def_type)
         D0                        D0
        struct DiskObject *GetDiskDefObject(LONG);

   FUNCTION
        This routine reads in a default Workbench disk object from disk.
        The valid def_types can be found in workbench/workbench.h and
        currently include WBDISK thru WBGARBAGE.  If the call fails,
        it will return zero.  The reason for the failure may be obtained
        via IoErr().

        Using this routine protects you from any future changes to
        the way default icons are stored within the system.

   INPUTS
        def_type - default icon type (WBDISK thru WBKICK).  Note that the
                   define 'WBDEVICE' is not currently supported.

   RESULTS
        diskobj -- the default Workbench disk object in question

   SEE ALSO
        PutDefDiskObject

   BUGS
        None
```

icon.library Page 9

```
icon.library/GetDiskObject                          icon.library/GetDiskObject

   NAME
       GetDiskObject - read in a Workbench disk object from disk.

   SYNOPSIS
       diskobj = GetDiskObject(name)
         D0                    A0

        struct DiskObject *GetDiskObject(char *);

   FUNCTION
        This routine reads in a Workbench disk object in from disk.  The
        name parameter will have a ".info" postpended to it, and the
        info file of that name will be read.  If the call fails,
        it will return zero.  The reason for the failure may be obtained
        via IoErr().

        Using this routine protects you from any future changes to
        the way icons are stored within the system.

        A FreeList structure is allocated just after the DiskObject
        structure; FreeDiskObject makes use of this to get rid of the
        memory that was allocated.

   INPUTS
        name -- name of the object (char *) or NULL if you just want a
                DiskObject structure allocated for you (useful when
                calling AddAppIcon in workbench.library).

   RESULTS
        diskobj -- the Workbench disk object in question

   SEE ALSO
        GetDiskObjectNew(), PutDiskObject(), DeleteDiskObject(),
        FreeDiskObject()

   BUGS
        None
```

icon.library Page 10

```
icon.library/GetDiskObjectNew                    icon.library/GetDiskObjectNew

   NAME
       GetDiskObjectNew - read in a Workbench disk object from disk.    (V36)

   SYNOPSIS
       diskobj = GetDiskObjectNew(name)
         D0                       A0

        struct DiskObject *GetDiskObject(char *);

   FUNCTION
        This routine reads in a Workbench disk object in from disk.  The
        name parameter will have a ".info" postpended to it, and the
        info file of that name will be read.  If the call fails,
        it will return zero.  The reason for the failure may be obtained
        via IoErr().

        Using this routine protects you from any future changes to
        the way icons are stored within the system.

        A FreeList structure is allocated just after the DiskObject
        structure; FreeDiskObject makes use of this to get rid of the
        memory that was allocated.

        This call is functionally identical to GetDiskObject with one exception.
        If its call to GetDiskObject fails, this function calls GetDefDiskObject.
        This is useful when there is no .info file for the icon you are trying
        to get a disk object for.  Applications that use workbench application
        windows MUST use this call if they want to handle the user dropping an
        icon (that doesn't have a .info file) on their window.  The V2.0
        icon editor program is an example of a workbench application window
        that uses this call.

   INPUTS
        name -- name of the object (char *) or NULL if you just want a
                DiskObject structure allocated for you (useful when
                calling AddAppIcon in workbench.library).

   RESULTS
        diskobj -- the Workbench disk object in question

   SEE ALSO
        FreeDiskObject(), GetDiskObject(), PutDiskObject(), DeleteDiskObject()

   BUGS
        None
```

```
icon.library/MatchToolValue                            icon.library/MatchToolValue

   NAME
       MatchToolValue - check a tool type variable for a particular value.

   SYNOPSIS
       result = MatchToolValue(typeString, value)
        D0                           A0        A1

        BOOL MatchToolValue(char *, char *);

   FUNCTION
        MatchToolValue is useful for parsing a tool type value for
        a known value.  It knows how to parse the syntax for a tool
        type value (in particular, it knows that '|' separates
        alternate values).  Note that the parsing is case insensitve.

   INPUTS
        typeString - a ToolType value (as returned by FindToolType)
        value - you are interested if value appears in typeString

   RESULTS
        result - TRUE if the value was in typeString else FALSE.

   EXAMPLE
        Assume there are two type strings:
            type1 = "text"
            type2 = "a|b|c"

        MatchToolValue( type1, "text" ) returns TRUE
        MatchToolValue( type1, "TEXT" ) returns TRUE
        MatchToolValue( type1, "data" ) returns FALSE
        MatchToolValue( type2, "a" ) returns TRUE
        MatchToolValue( type2, "b" ) returns TRUE
        MatchToolValue( type2, "d" ) returns FALSE
        MatchToolValue( type2, "a|b" ) returns FALSE

   SEE ALSO
        FindToolType()

   BUGS
        None
```

```
icon.library/PutDefDiskObject                    icon.library/PutDefDiskObject

   NAME
        PutDefDiskObject - write disk object as the default for its type.  (V36)

   SYNOPSIS
       status = PutDefDiskObject(diskobj)
        D0                         A0
        BOOL PutDefDiskObject(struct DiskObject *);

   FUNCTION
        This routine writes out a DiskObject structure, and its
        associated information.  If the call fails, a zero will
        be returned.  The reason for the failure may be obtained
        via IoErr().

        Note that this function calls PutDiskObject internally which means
        that this call (if sucessful) notifies workbench than an icon has
        been created/modified.

        Using this routine protects you from any future changes to
        the way default icons are stored within the system.

   INPUTS
        diskobj -- a pointer to a DiskObject

   RESULTS
        status -- TRUE if the call succeeded else FALSE

   SEE ALSO
        GetDefDiskObject

   BUGS
        None
```

```
icon.library/PutDiskObject                         icon.library/PutDiskObject

   NAME
       PutDiskObject - write out a DiskObject to disk.

   SYNOPSIS
       status = PutDiskObject(name, diskobj)
         D0                    A0      A1

        BOOL PutDiskObject(char *, struct DiskObject *);

   FUNCTION
        This routine writes out a DiskObject structure, and its
        associated information.  The file name of the info
        file will be the name parameter with a
        ".info" postpended to it.  If the call fails, a zero will
        be returned.  The reason for the failure may be obtained
        via IoErr().

        As of release V2.0, PutDiskObject (if sucessful) notifies workbench
        han an icon has been created/modified.

        Using this routine protects you from any future changes to
        the way icons are stored within the system.

   INPUTS
        name -- name of the object (pointer to a character string)
        diskobj -- a pointer to a DiskObject

   RESULTS
        status -- TRUE if the call succeeded else FALSE

   NOTES
        It is recommended that if you wish to copy an icon from one place
        to another than you use GetDiskObject() and PutDiskObject()
        and do not copy them directly.

   SEE ALSO
        GetDiskObject(), FreeDiskObject(), DeleteDiskObject()

   BUGS
        None
```

TABLE OF CONTENTS

iffparse.library/AllocIFF
iffparse.library/AllocLocalItem
iffparse.library/CloseClipboard
iffparse.library/CloseIFF
iffparse.library/CollectionChunk
iffparse.library/CollectionChunks
iffparse.library/CurrentChunk
iffparse.library/EntryHandler
iffparse.library/ExitHandler
iffparse.library/FindCollection
iffparse.library/FindLocalItem
iffparse.library/FindProp
iffparse.library/FindPropContext
iffparse.library/FreeIFF
iffparse.library/FreeLocalItem
iffparse.library/GoodID
iffparse.library/GoodType
iffparse.library/HookEntry
iffparse.library/IDtoStr
iffparse.library/InitIFF
iffparse.library/InitIFFasClip
iffparse.library/InitIFFasDOS
iffparse.library/LocalItemData
iffparse.library/OpenClipboard
iffparse.library/OpenIFF
iffparse.library/ParentChunk
iffparse.library/ParseIFF
iffparse.library/PopChunk
iffparse.library/PropChunk
iffparse.library/PropChunks
iffparse.library/PushChunk
iffparse.library/ReadChunkBytes
iffparse.library/ReadChunkRecords
iffparse.library/SetLocalItemPurge
iffparse.library/StopChunk
iffparse.library/StopChunks
iffparse.library/StopOnExit
iffparse.library/StoreItemInContext
iffparse.library/StoreLocalItem
iffparse.library/WriteChunkBytes
iffparse.library/WriteChunkRecords

```
iffparse.library/AllocIFF                              iffparse.library/AllocIFF

   NAME
        AllocIFF -- Create a new IFFHandle structure.

   SYNOPSIS
        iff = AllocIFF ()
        d0

        struct IFFHandle *iff;

   FUNCTION
        Allocates a new IFFHandle structure and initializes the basic values.
        This function is the only supported way to create an IFFHandle
        structure since there are private fields that need to be initialized.

   INPUTS

   RESULT
        iff     - pointer to IFFHandle structure or NULL if the allocation
                  failed.

   EXAMPLE

   NOTES

   BUGS

   SEE ALSO
        FreeIFF()
```

```
iffparse.library/AllocLocalItem                   iffparse.library/AllocLocalItem

   NAME
        AllocLocalItem -- Create a local context item structure.

   SYNOPSIS
        item = AllocLocalItem (type, id, ident, usize)
         d0                     d0   d1   d2     d3

        struct LocalContextItem *item;
        LONG                    type, id, ident, usize;

   FUNCTION
        Allocates and initializes a LocalContextItem structure with "usize"
        bytes of associated user data.  This is the only supported way to
        create such an item.  The user data can be accessed with the
        LocalItemData function.  An item created with this function
        automatically has its purge vectors set up correctly to dispose of
        itself and its associated user data area.  Any additional cleanup
        should be done with a user-supplied purge vector.

   INPUTS
        type,id - additional longword identification values.
        ident   - longword identifier for class of context item.
        usize   - number of bytes of user data to allocate for this item.

   RESULT
        item    - pointer to initialized LocalContextItem or NULL if the
                  allocation failed.

   EXAMPLE

   NOTES

   BUGS

   SEE ALSO
        FreeLocalItem(), LocalItemData(), StoreLocalItem(),
        StoreItemInContext(), SetLocalItemPurge()
```

```
iffparse.library/CloseClipboard                   iffparse.library/CloseClipboard

   NAME
        CloseClipboard -- Close and free an open ClipboardHandle.

   SYNOPSIS
        CloseClipboard (clip)
                        a0

        struct ClipboardHandle *clip;

   FUNCTION
        Closes the clipboard.device and frees the ClipboardHandle structure.

   INPUTS
        clip    - pointer to ClipboardHandle struct created with
                  OpenClipboard.

   RESULT

   EXAMPLE

   NOTES

   BUGS

   SEE ALSO
        OpenClipboard(), InitIFFasClip()
```

iffparse.library Page 5

```
iffparse.library/CloseIFF                              iffparse.library/CloseIFF

   NAME
       CloseIFF -- Close an IFF context.

   SYNOPSIS
       CloseIFF (iff)
                 a0

       struct IFFHandle *iff;

   FUNCTION
       Completes an IFF read or write operation by closing the IFF context
       established for this IFFHandle struct.  The IFFHandle struct itself
       is left ready for re-use and a new context can be opened with
       OpenIFF().  This function can be used for cleanup if a read or write
       fails partway through.

       As part of its cleanup operation, CloseIFF() calls the client-
       supplied stream hook vector.  The IFFStreamCmd packet will be set
       as follows:

               sc_Command:     IFFCMD_CLEANUP
               sc_Buf:         (Not applicable)
               sc_NBytes:      (Not applicable)

       This operation is NOT permitted to fail;  any error code returned
       will be ignored (best to return 0, though).  DO NOT write to this
       structure.

   INPUTS
       iff     - pointer to IFFHandle struct previously opened with
                 OpenIFF().

   RESULT

   EXAMPLE

   NOTES

   BUGS

   SEE ALSO
       OpenIFF(), InitIFF()
```

iffparse.library Page 6

```
iffparse.library/CollectionChunk                iffparse.library/CollectionChunk

   NAME
       CollectionChunk -- declare a chunk type for collection.

   SYNOPSIS
       error = CollectionChunk (iff, type, id)
        d0                       a0    d0   d1

       LONG              error;
       struct IFFHandle *iff;
       LONG              type;
       LONG              id;

   FUNCTION
       Installs an entry handler for chunks with the given type and id so
       that the contents of those chunks will be stored as they are
       encountered.  This is like PropChunk() except that more than one
       chunk of this type can be stored in lists which can be returned by
       FindCollection().  The storage of these chunks still follows the
       property chunk scoping rules for IFF files so that at any given
       point, stored collection chunks will be valid in the current context.

   INPUTS
       iff     - pointer to IFFHandle struct (does not need to be open).
       type    - type code for the chunk to declare (ex. "ILBM").
       id      - identifier for the chunk to declare (ex. "CRNG").

   RESULT
       error   - 0 if successful or an IFFERR_#? error code if not
                 successful.

   EXAMPLE

   NOTES

   BUGS

   SEE ALSO
       CollectionChunks(), FindCollection(), PropChunk()
```

```
iffparse.library/CollectionChunks          iffparse.library/CollectionChunks

   NAME
        CollectionChunks -- Declare many collection chunks at once.

   SYNOPSIS
        error = CollectionChunks (iff, list, n)
         d0                       a0    a1  d0

        LONG             error;
        struct IFFHandle *iff;
        LONG             *list;
        LONG             n;

   FUNCTION
        Declares multiple collection chunks from a list.  The list argument
        is a pointer to an array of long words arranged in pairs.  The format
        for the list is as follows:

                TYPE1, ID1, TYPE2, ID2, ..., TYPEn, IDn

        The argument n is the number of pairs.  CollectionChunks() just calls
        CollectionChunk() n times.

   INPUTS
        iff     - pointer to IFFHandle struct.
        list    - pointer to array of longword chunk types and identifiers.
        n       - number of chunks to declare.

   RESULT
        error   - 0 if successful or an IFFERR_#? error code if not
                  successful.

   EXAMPLE

   NOTES

   BUGS

   SEE ALSO
        CollectionChunk()
```

```
iffparse.library/CurrentChunk                    iffparse.library/CurrentChunk

   NAME
        CurrentChunk -- Get context node for current chunk.

   SYNOPSIS
        top = CurrentChunk (iff)
        d0                  a0

        struct ContextNode      *top;
        struct IFFHandle        *iff;

   FUNCTION
        Returns top context node for the given IFFHandle struct.  The top
        context node corresponds to the chunk most recently pushed on the
        stack, which is the chunk where the stream is currently positioned.
        The ContextNode structure contains information on the type of chunk
        currently being parsed (or written), its size and the current
        position within the chunk.

   INPUTS
        iff     - pointer to IFFHandle struct.

   RESULT
        top     - pointer to top context node or NULL if none.

   EXAMPLE

   NOTES

   BUGS

   SEE ALSO
        PushChunk(), PopChunk(), ParseIFF(), ParentChunk()
```

```
iffparse.library/EntryHandler                    iffparse.library/EntryHandler

   NAME
       EntryHandler -- Add an entry handler to the IFFHandle context.

   SYNOPSIS
       error = EntryHandler (iff, type, id, position, hook, object)
        d0                   a0    d0   d1    d2       a1     a2

       LONG             error;
       struct IFFHandle *iff;
       LONG             type, id, position;
       struct Hook      *hook;
       APTR             object;

   FUNCTION
       Installs an entry handler vector for a specific type of chunk into
       the context for the given IFFHandle struct.  Type and id are the
       longword identifiers for the chunk to handle.  The hook is a client-
       supplied standard 2.0 Hook structure, properly initialized.  Position
       tells where to put the handler in the context.  The handler will be
       called whenever the parser enters a chunk of the given type, so the
       IFF stream will be positioned to read the first data byte in the
       chunk.  The handler will execute in the same context as whoever
       called ParseIFF().  The handler will be called (through the hook)
       with the following arguments:

               A0:     the Hook pointer you passed.
               A2:     the 'object' pointer you passed.
               A1:     pointer to a LONG containing the value
                       IFFCMD_ENTRY.

       The error code your call-back routine returns will affect the parser
       in three different ways:

       Return value            Result
       ------------            ------
       0:                      Normal success;  ParseIFF() will continue
                               through the file.
       IFF_RETURN2CLIENT:      ParseIFF() will stop and return the value 0.
                               (StopChunk() is internally implemented using
                               this return value.)
       Any other value:        ParseIFF() will stop and return the value
                               you supplied.  This is how errors should be
                               returned.

   INPUTS
       iff      - pointer to IFFHandle struct.
       type     - type code for chunk to handle (ex. "ILBM").
       id       - ID code for chunk to handle (ex. "CMAP").
       position- Local context item position.  One of the IFFSLI_#? codes.
       hook     - pointer to Hook structure.
       object   - a client-defined pointer which is passed in A2 during call-
                  back.

   RESULT
       error    - 0 if successful or an IFFERR_#? error code if not
                  successful.

   EXAMPLE

   NOTES

   BUGS
       Returning the values IFFERR_EOF or IFFERR_EOC from the call-back
       routine *may* confuse the parser.
```

```
       There is no way to explicitly remove a handler once installed.
       However, by installing a do-nothing handler using IFFSLI_TOP,
       previous handlers will be overridden until the context expires.

   SEE ALSO
       ExitHandler(), StoreLocalItem(), StoreItemInContext(),
       utility/hooks.h
```

```
iffparse.library/ExitHandler                     iffparse.library/ExitHandler

   NAME
       ExitHandler -- Add an exit handler to the IFFHandle context.

   SYNOPSIS
       error = ExitHandler (iff, type, id, position, hook, object)
        d0                  a0    d0   d1    d2        a1     a2

       LONG             error;
       struct IFFHandle *iff;
       LONG             type, id, position;
       struct Hook      *hook;
       APTR             object;

   FUNCTION
       Installs an exit handler vector for a specific type of chunk into the
       context for the given IFFHandle struct.  Type and id are the longword
       identifiers for the chunk to handle.  The hook is a client-supplied
       standard 2.0 Hook structure, properly initialized.  Position tells
       where to put the handler in the context.  The handler will be called
       just before the parser exits the given chunk in the "pause" parse
       state.  The IFF stream may not be positioned predictably within the
       chunk.  The handler will execute in the same context as whoever
       called ParseIFF().  The handler will be called (through the hook)
       with the following arguments:

               A0:     the Hook pointer you passed.
               A2:     the 'object' pointer you passed.
               A1:     pointer to a LONG containing the value
                       IFFCMD_EXIT.

       The error code your call-back routine returns will affect the parser
       in three different ways:

       Return value            Result
       ------------            ------
       0:                      Normal success;  ParseIFF() will continue
                               through the file.
       IFF_RETURN2CLIENT:      ParseIFF() will stop and return the value 0.
                               (StopChunk() is internally implemented using
                               this return value.)
       Any other value:        ParseIFF() will stop and return the value
                               you supplied.  This is how errors should be
                               returned.

   INPUTS
       iff     - pointer to IFFHandle struct.
       type    - type code for chunk to handle (ex. "ILBM").
       id      - identifier code for chunk to handle (ex. "CMAP").
       position- local context item position.  One of the IFFSLI_#? codes.
       hook    - pointer to Hook structure.
       object  - a client-defined pointer which is passed in A2 during call-
                 back.

   RESULT
       error   - 0 if successful or an IFFERR_#? error code if not
                 successful.

   EXAMPLE

   NOTES

   BUGS
       Returning the values IFFERR_EOF or IFFERR_EOC from the call-back
       routine *may* confuse the parser.
```

```
       There is no way to explicitly remove a handler once installed.
       However, by installing a do-nothing handler using IFFSLI_TOP,
       previous handlers will be overridden until the context expires.

   SEE ALSO
       EntryHandler(), StoreLocalItem(), StoreItemInContext(),
       utility/hooks.h
```

iffparse.library Page 13

```
iffparse.library/FindCollection               iffparse.library/FindCollection

   NAME
        FindCollection -- Get a pointer to the current list of collection
                          items.

   SYNOPSIS
        ci = FindCollection (iff, type, id)
        d0                   a0   d0    d1

        struct CollectionItem   *ci;
        struct IFFHandle        *iff;
        LONG                    type, id;

   FUNCTION
        Returns a pointer to a list of CollectionItem structures for each of
        the collection chunks of the given type encountered so far in the
        course of parsing this IFF file.  The items appearing first in the
        list will be the ones encountered most recently.

   INPUTS
        iff     - pointer to IFFHandle struct.
        type    - type code to search for.
        id      - identifier code to search for.

   RESULT
        ci      - pointer to last collection chunk encountered with
                  links to previous ones.

   EXAMPLE

   NOTES

   BUGS

   SEE ALSO
        CollectionChunk(), CollectionChunks()
```

iffparse.library Page 14

```
iffparse.library/FindLocalItem                 iffparse.library/FindLocalItem

   NAME
        FindLocalItem -- Return a local context item from the context stack.

   SYNOPSIS
        lci = FindLocalItem (iff, type, id, ident)
        d0                   a0   d0    d1  d2

        struct LocalContextItem *lci;
        struct IFFHandle        *iff;
        LONG                    type, id, ident;

   FUNCTION
        Searches the context stack of the given IFFHandle struct for a local
        context item which matches the given ident, type and id.  This
        function searches the context stack from the most current context
        backwards, so that the item found (if any) will be the one with
        greatest precedence in the context stack.

   INPUTS
        iff     - pointer to IFFHandle struct.
        type    - type code to search for.
        id      - ID code to search for.
        ident   - ident code for the class of context item to search for
                  (ex. "exhd" -- exit handler).

   RESULT
        lci     - pointer local context item if found, or NULL if nothing
                  matched.

   EXAMPLE

   NOTES

   BUGS
        It really should have some sort of wildcarding capability.

   SEE ALSO
        StoreLocalItem()
```

```
iffparse.library/FindProp                              iffparse.library/FindProp

   NAME
        FindProp -- Search for a stored property chunk.

   SYNOPSIS
        sp = FindProp (iff, type, id)
        d0              a0    d0   d1

        struct StoredProperty    *sp;
        struct IFFHandle         *iff;
        LONG                     type, id;

   FUNCTION
        Searches for the stored property which is valid in the given context.
        Property chunks are automatically stored by ParseIFF() when
        pre-declared by PropChunk() or PropChunks().  The StoredProperty
        struct, if found, contains a pointer to a data buffer containing the
        contents of the stored property.

   INPUTS
        iff     - pointer to IFFHandle struct.
        type    - type code for chunk to search for (ex. "ILBM").
        id      - identifier code for chunk to search for (ex. "CMAP").

   RESULT
        sp      - pointer to stored property, if found, or NULL if none
                  found.

   EXAMPLE

   NOTES

   BUGS

   SEE ALSO
        PropChunk(), PropChunks()
```

```
iffparse.library/FindPropContext                iffparse.library/FindPropContext

   NAME
        FindPropContext -- Get the property context for the current state.

   SYNOPSIS
        cn = FindPropContext (iff)
        d0                    a0

        struct ContextNode      *cn;
        struct IFFHandle        *iff;

   FUNCTION
        Locates the context node which would be the scoping chunk for
        properties in the current parsing state.  (Huh?)  This is used for
        locating the proper scoping context for property chunks i.e. the
        scope from which a property would apply.  This is usually the FORM
        or LIST with the highest precedence in the context stack.

        If you don't understand this, read the IFF spec a couple more times.

   INPUTS
        iff     - pointer to IFFHandle struct.

   RESULT
        cn      - ContextNode of property scoping chunk.

   EXAMPLE

   NOTES

   BUGS

   SEE ALSO
        CurrentChunk(), ParentChunk(), StoreItemInContext()
```

iffparse.library Page 17

```
iffparse.library/FreeIFF                                iffparse.library/FreeIFF

   NAME
        FreeIFF -- Deallocate an IFFHandle struct.

   SYNOPSIS
        FreeIFF (iff)
                 a0

        struct IFFHandle *iff;

   FUNCTION
        Deallocates all resources associated with this IFFHandle struct.  The
        struct MUST have already been closed with CloseIFF().

   INPUTS
        iff     - pointer to IFFHandle struct to free.

   RESULT

   EXAMPLE

   NOTES

   BUGS

   SEE ALSO
        AllocIFF(), CloseIFF()
```

iffparse.library Page 18

```
iffparse.library/FreeLocalItem                    iffparse.library/FreeLocalItem

   NAME
        FreeLocalItem -- Deallocate a local context item structure.

   SYNOPSIS
        FreeLocalItem (lci)
                       a0

        struct LocalContextItem *lci;

   FUNCTION
        Frees the memory for the local context item and any associated user
        memory as allocated with AllocLocalItem.  User purge vectors should
        call this function after they have freed any other resources
        associated with this item.

        Note that FreeLocalItem() does NOT call the custom purge vector set
        up through SetLocalItemPurge(); all it does is free the local context
        item.  (This implies that your custom purge vector would want to call
        this to ultimately free the LocalContextItem.)  (This description
        still seems muddy; how to clear it up?)

   INPUTS
        lci     - pointer to LocalContextItem created with AllocLocalItem.

   RESULT

   EXAMPLE

   NOTES

   BUGS

   SEE ALSO
        AllocLocalItem()
```

```
iffparse.library/GoodID                                 iffparse.library/GoodID

   NAME
       GoodID -- Test if an identifier follows the IFF 85 specification.

   SYNOPSIS
       isok = GoodID (id)
        d0            d0

       LONG isok, id;

   FUNCTION
       Tests the given longword identifier to see if it meets all the EA IFF
       85 specifications for a chunk ID.  If so, it returns non-zero,
       otherwise 0.

   INPUTS
       id       - potential 32 bit identifier.

   RESULT
       isok     - non-zero if this is a valid ID, 0 otherwise.

   EXAMPLE

   NOTES

   BUGS

   SEE ALSO
       GoodType()
```

```
iffparse.library/GoodType                             iffparse.library/GoodType

   NAME
       GoodType -- Test if a type follows the IFF 85 specification.

   SYNOPSIS
       isok = GoodType (type)
        d0              d0

       LONG isok, type;

   FUNCTION
       Tests the given longword type identifier to see if it meets all the
       EA IFF 85 specifications for a FORM type (requirements for a FORM
       type are more stringent than those for a simple chunk ID).  If it
       complies, GoodType() returns non-zero, otherwise 0.

   INPUTS
       type     - potential 32 bit format type identifier.

   RESULT
       isok     - non-zero if this is a valid type id, 0 otherwise.

   EXAMPLE

   NOTES

   BUGS

   SEE ALSO
       GoodID()
```

iffparse.library Page 21

```
iffparse.library/HookEntry                    iffparse.library/HookEntry


   NAME
       HookEntry -- call-back stub vector (LANGUAGE SPECIFIC LINK ROUTINE)

   SYNOPSIS
       This function is never called directly by the client.

   FUNCTION
       HookEntry's purpose is to do language-specific setup and conversion
       of parameters passed from a library to a client call-back routine.
       Under Kickstart 2.0, a standard for call-backs has been established.
       The registers will contain the following items:

               A0:     pointer to hook that enabled us to get here.
               A2:     pointer to "object."
               A1:     pointer to "message packet."

       In iffparse, the "object" will vary from routine to routine.  The
       "message packet" is also specific to the operation involved (RTFM!).

       THIS ROUTINE IS NOT PART OF IFFPARSE.  It, or something similar, is
       part of the compiler vendor's link library.  (If it's not there,
       cobbling up your own isn't too hard.)

   SEE ALSO
       EntryHandler(), ExitHandler(), InitIFF(), SetLocalItemPurge(),
       utility/hooks.h (A must-read;  LOTS of details in there)
```

iffparse.library Page 22

```
iffparse.library/IDtoStr                                iffparse.library/IDtoStr

   NAME
       IDtoStr -- Convert a longword identifier to a null-terminated string.

   SYNOPSIS
       str = IDtoStr (id, buf)
       d0             d0  a0

       STRPTR  str;
       LONG    id;
       STRPTR  buf;

   FUNCTION
       Writes the ASCII equivalent of the given longword ID into buf as a
       null-terminated string.

   INPUTS
       id      - longword ID.
       buf     - character buffer to accept string (at least 5 chars).

   RESULT
       str     - the value of 'buf'.

   EXAMPLE

   NOTES

   BUGS

   SEE ALSO
```

```
iffparse.library/InitIFF                             iffparse.library/InitIFF

   NAME
       InitIFF -- Initialize an IFFHandle struct as a user stream.

   SYNOPSIS
       InitIFF (iff, flags, streamhook)
                a0   d0      a1

       struct IFFHandle *iff;
       LONG             flags;
       struct Hook      *streamhook;

   FUNCTION
       Initializes an IFFHandle as a general user-defined stream by
       allowing the user to declare a hook that the library will call to
       accomplish the low-level reading, writing, and seeking of the stream.
       Flags are the stream I/O flags for the specified stream; typically a
       combination of the IFFF_?SEEK flags.

       The stream vector is called with the following arguments:

               A0:     pointer to streamhook.
               A2:     pointer to IFFHandle struct.
               A1:     pointer to IFFStreamCmd struct.

       The IFFStreamCmd packet appears as follows:

               sc_Command:     Contains an IFFCMD_#? value
               sc_Buf:         Pointer to memory buffer
               sc_NBytes:      Number of bytes involved in operation

       The values taken on by sc_Command, and their meaning, are as follows:

       IFFCMD_INIT:
               Prepare your stream for reading.  This is used for certain
               streams that can't be read immediately upon opening, and need
               further preparation.  (The clipboard.device is an example of
               such a stream.)  This operation is allowed to fail;  any
               error code will be returned directly to the client.  sc_Buf
               and sc_NBytes have no meaning here.
       IFFCMD_CLEANUP:
               Terminate the transaction with the associated stream.  This
               is used with streams that can't simply be closed.  (Again,
               the clipboard is an example of such a stream.)  This
               operation is not permitted to fail;  any error returned will
               be ignored (best to return 0, though).  sc_Buf and sc_NBytes
               have no meaning here.
       IFFCMD_READ:
               Read from the stream.  You are to read sc_NBytes from the
               stream and place them in the buffer pointed to by sc_Buf.
               Any (non-zero) error returned will be remapped by the parser
               into IFFERR_READ.
       IFFCMD_WRITE:
               Write to the stream.  You are to write sc_NBytes to the
               stream from the buffer pointed to by sc_Buf.  Any (non-zero)
               error returned will be remapped by the parser into
               IFFERR_WRITE.
       IFFCMD_SEEK:
               Seek on the stream.  You are to perform a seek on the stream
               relative to the current position.  sc_NBytes is signed;
               negative values mean seek backward, positive values mean seek
               forward.  sc_Buf has no meaning here.  Any (non-zero) error
               returned will be remapped by the parser into IFFERR_SEEK.

       All errors are returned in D0.  A return of 0 indicates success.
```

```
       UNDER NO CIRCUMSTANCES are you permitted to write to the IFFStreamCmd
       structure.

   INPUTS
       iff     - pointer to IFFHandle structure to initialize.
       flags   - stream I/O flags for the IFFHandle.
       hook    - pointer to Hook structure.

   RESULT

   EXAMPLE

   NOTES

   BUGS

   SEE ALSO
       utility/hooks.h
```

```
iffparse.library/InitIFFasClip                  iffparse.library/InitIFFasClip

   NAME
        InitIFFasClip -- Initialize an IFFHandle as a clipboard stream.

   SYNOPSIS
        InitIFFasClip (iff)
                       a0

        struct IFFHandle *iff;

   FUNCTION
        Initializes the given IFFHandle to be a clipboard stream.  The
        function initializes the stream processing vectors to operate on
        streams of the ClipboardHandle type.  The iff_Stream field will still
        need to be initialized to point to a ClipboardHandle as returned from
        OpenClipboard().

   INPUTS
        iff     - pointer to IFFHandle struct.

   RESULT

   EXAMPLE

   NOTES

   BUGS

   SEE ALSO
        OpenClipboard()
```

```
iffparse.library/InitIFFasDOS                    iffparse.library/InitIFFasDOS

   NAME
        InitIFFasDOS -- Initialize an IFFHandle as a DOS stream.

   SYNOPSIS
        InitIFFasDOS (iff)
                      a0

        struct IFFHandle *iff;

   FUNCTION
        The function initializes the given IFFHandle to operate on DOS
        streams.  The iff_Stream field will need to be initialized as a BPTR
        returned from the DOS function Open().

   INPUTS
        iff     - pointer to IFFHandle struct.

   RESULT

   EXAMPLE

   NOTES

   BUGS

   SEE ALSO
```

```
iffparse.library/LocalItemData                    iffparse.library/LocalItemData

   NAME
       LocalItemData -- Get pointer to user data for local context item.

   SYNOPSIS
       data = LocalItemData (lci)
        d0                   a0

       UBYTE                  *data;
       struct LocalContextItem *lci;

   FUNCTION
       Returns pointer to the user data associated with the given local
       context item.  The size of the data area depends on the "usize"
       argument used when allocating this item.  If the pointer to the item
       given (lci) is NULL, the function also returns NULL.

   INPUTS
       lci     - pointer to local context item or NULL.

   RESULT
       data    - pointer to user data area or NULL if lci is NULL.

   EXAMPLE

   NOTES

   BUGS
       Currently, there is no way to determine the size of the user data
       area; you have to 'know'.

   SEE ALSO
       AllocLocalItem(), FreeLocalItem()
```

```
iffparse.library/OpenClipboard                    iffparse.library/OpenClipboard

   NAME
       OpenClipboard -- Create a handle on a clipboard unit.

   SYNOPSIS
       ch = OpenClipboard (unit)
       d0                  d0

       struct ClipboardHandle  *ch;
       LONG                    unit;

   FUNCTION
       Opens the clipboard.device and opens a stream for the specified unit
       (usually PRIMARY_CLIP).  This handle structure will be used as the
       clipboard stream for IFFHandles initialized as clipboard streams by
       InitIFFasClip().

   INPUTS
       unit    - clipboard unit number (usually PRIMARY_CLIP).

   RESULT
       ch      - pointer to ClipboardHandle structure or NULL if
                 unsuccessful.

   EXAMPLE

   NOTES

   BUGS

   SEE ALSO
       InitIFFasClip(), CloseClipboard()
```

iffparse.library Page 29

```
iffparse.library/OpenIFF                               iffparse.library/OpenIFF

   NAME
       OpenIFF -- Prepare an IFFHandle to read or write a new IFF stream.

   SYNOPSIS
       error = OpenIFF (iff, rwmode)
        d0              a0   d0

       LONG            error;
       struct IFFHandle *iff;
       LONG            rwmode;

   FUNCTION
       Initializes an IFFHandle struct for a new read or write.  The
       direction of the I/O is given by the value of rwmode, which can be
       either IFFF_READ or IFFF_WRITE.

       As part of its initialization procedure, OpenIFF() calls the client-
       supplied stream hook vector.  The IFFStreamCmd packet will contain
       the following:

               sc_Command:     IFFCMD_INIT
               sc_Buf:         (Not applicable)
               sc_NBytes:      (Not applicable)

       This operation is permitted to fail.  DO NOT write to this structure.

   INPUTS
       iff     - pointer to IFFHandle struct.
       rwmode  - IFFF_READ or IFFF_WRITE

   RESULT
       error   - contains an error code or 0 if successful.

   EXAMPLE

   NOTES

   BUGS

   SEE ALSO
       CloseIFF(), InitIFF()
```

iffparse.library Page 30

```
iffparse.library/ParentChunk                       iffparse.library/ParentChunk

   NAME
       ParentChunk -- Get the nesting context node for the given chunk.

   SYNOPSIS
       parent = ParentChunk (cn)
         d0                  a0

       struct ContextNode *parent, *cn;

   FUNCTION
       Returns a context node for the chunk containing the chunk for the
       given context node.  This function effectively moves down the context
       stack into previously pushed contexts.  For example, to get a
       ContextNode pointer for the enclosing FORM chunk while reading a data
       chunk, use: ParentChunk (CurrentChunk (iff)) to find this pointer.
       The ContextNode structure contains information on the type of chunk
       and its size.

   INPUTS
       cn      - pointer to a context node.

   RESULT
       parent  - pointer to the enclosing context node or NULL if none.

   EXAMPLE

   NOTES

   BUGS

   SEE ALSO
       CurrentChunk()
```

```
iffparse.library/ParseIFF                          iffparse.library/ParseIFF

   NAME
        ParseIFF -- Parse an IFF file from an IFFHandle struct stream.

   SYNOPSIS
        error = ParseIFF (iff, control)
         d0                a0     d0

        LONG             error;
        struct IFFHandle *iff;
        LONG             control;

   FUNCTION
        This is the biggie.

        Traverses a file opened for read by pushing chunks onto the context
        stack and popping them off directed by the generic syntax of IFF
        files.  As it pushes each new chunk, it searches the context stack
        for handlers to apply to chunks of that type.  If it finds an entry
        handler it will invoke it just after entering the chunk.  If it finds
        an exit handler it will invoke it just before leaving the chunk.
        Standard handlers include entry handlers for pre-declared
        property chunks and collection chunks and entry and exit handlers for
        for stop chunks - that is, chunks which will cause the ParseIFF()
        function to return control to the client.  Client programs can also
        provide their own custom handlers.

        The control flag can have three values:

        IFFPARSE_SCAN:
                In this normal mode, ParseIFF() will only return control to
                the caller when either:
                        1) an error is encountered,
                        2) a stop chunk is encountered, or a user handler
                           returns the special IFF_RETURN2CLIENT code, or
                        3) the end of the logical file is reached, in which
                           case IFFERR_EOF is returned.

                ParseIFF() will continue pushing and popping chunks until one
                of these conditions occurs.  If ParseIFF() is called again
                after returning, it will continue to parse the file where it
                left off.

        IFFPARSE_STEP and _RAWSTEP:
                In these two modes, ParseIFF() will return control to the
                caller after every step in the parse, specifically, after
                each push of a context node and just before each pop.  If
                returning just before a pop, ParseIFF() will return
                IFFERR_EOC, which is not an error, per se, but is just an
                indication that the most recent context is ending.  In STEP
                mode, ParseIFF() will invoke the handlers for chunks, if
                any, before returning.  In RAWSTEP mode, ParseIFF() will not
                invoke any handlers and will return right away.  In both
                cases the function can be called multiple times to step
                through the parsing of the IFF file.

   INPUTS
        iff     - pointer to IFFHandle struct.
        control - control code (IFFPARSE_SCAN, _STEP or _RAWSTEP).

   RESULT
        error   - 0 or IFFERR_#? value or return value from user handler.

   EXAMPLE
```

```
   NOTES

   BUGS

   SEE ALSO
        PushChunk(), PopChunk(), EntryHandler(), ExitHandler(),
        PropChunk[s](), CollectionChunk[s](), StopChunk(), StopOnExit()
```

iffparse.library Page 33

```
iffparse.library/PopChunk                              iffparse.library/PopChunk

   NAME
       PopChunk -- Pop top context node off context stack.

   SYNOPSIS
       error = PopChunk (iff)
        d0                a0

       LONG             error;
       struct IFFHandle *iff;

   FUNCTION
       Pops top context chunk and frees all associated local context items.
       The function is normally called only for writing files and signals
       the end of a chunk.

   INPUTS
       iff      - pointer to IFFHandle struct.

   RESULT
       error    - 0 if successful or an IFFERR_#? error code if not
                  successful.

   EXAMPLE

   NOTES

   BUGS

   SEE ALSO
       PushChunk()
```

iffparse.library Page 34

```
iffparse.library/PropChunk                            iffparse.library/PropChunk

   NAME
       PropChunk -- Specify a property chunk to store.

   SYNOPSIS
       error = PropChunk (iff, type, id)
        d0                 a0    d0   d1

       LONG             error;
       struct IFFHandle *iff;
       LONG             type;
       LONG             id;

   FUNCTION
       Installs an entry handler for chunks with the given type and ID so
       that the contents of those chunks will be stored as they are
       encountered.  The storage of these chunks follows the property chunk
       scoping rules for IFF files so that at any given point, a stored
       property chunk returned by FindProp() will be the valid property for
       the current context.

   INPUTS
       iff      - pointer to IFFHandle struct (does not need to be open).
       type     - type code for the chunk to declare (ex. "ILBM").
       id       - identifier for the chunk to declare (ex. "CMAP").

   RESULT
       error    - 0 if successful or an IFFERR_#? error code if not
                  successful.

   EXAMPLE

   NOTES

   BUGS

   SEE ALSO
       PropChunks(), FindProp(), CollectionChunk()
```

```
iffparse.library/PropChunks                        iffparse.library/PropChunks

   NAME
        PropChunks -- Declare many property chunks at once.

   SYNOPSIS
        error = PropChunks (iff, list, n)
         d0                   a0    a1  d0

        LONG             error;
        struct IFFHandle *iff;
        LONG             *list;
        LONG             n;

   FUNCTION
        Declares multiple property chunks from a list.  The list argument is
        a pointer to an array of long words arranged in pairs, and has the
        following format:

                TYPE1, ID1, TYPE2, ID2, ..., TYPEn, IDn

        The argument n is the number of pairs.  PropChunks() just calls
        PropChunk() n times.

   INPUTS
        iff     - pointer to IFFHandle struct.
        list    - pointer to array of longword chunk types and identifiers.
        n       - number of chunks to declare.

   RESULT
        error   - 0 if successful or an IFFERR_#? error code if not
                  successful.

   EXAMPLE

   NOTES

   BUGS

   SEE ALSO
        PropChunk()
```

```
iffparse.library/PushChunk                          iffparse.library/PushChunk

   NAME
        PushChunk -- Push a new context node on the context stack.

   SYNOPSIS
        error = PushChunk (iff, type, id, size)
         d0                  a0    d0   d1   d2

        LONG             error;
        struct IFFHandle *iff;
        LONG             type, id, size;

   FUNCTION
        Pushes a new context node on the context stack by reading it from the
        stream if this is a read file, or by creating it from the passed
        parameters if this is a write file.  Normally this function is only
        called in write mode, where the type and id codes specify the new
        chunk to create.  If this is a leaf chunk, i.e. a local chunk inside
        a FORM or PROP chunk, then the type argument is ignored.  If the size
        is specified then the chunk writing functions will enforce this size.
        If the size is given as IFFSIZE_UNKNOWN, the chunk will expand to
        accommodate whatever is written into it.

   INPUTS
        iff     - pointer to IFFHandle struct.
        type    - chunk type specifier (ex. ILBM) (ignored for read mode or
                  leaf chunks).
        id      - chunk id specifier (ex. CMAP) (ignored for read mode).
        size    - size of the chunk to create or IFFSIZE_UNKNOWN (ignored for
                  read mode).

   RESULT
        error   - 0 if successful or an IFFERR_#? error code if not
                  successful.

   EXAMPLE

   NOTES

   BUGS

   SEE ALSO
        PopChunk(), WriteChunkRecords(), WriteChunkBytes()
```

iffparse.library Page 37

```
iffparse.library/ReadChunkBytes                    iffparse.library/ReadChunkBytes

   NAME
        ReadChunkBytes -- Read bytes from the current chunk into a buffer.

   SYNOPSIS
        actual = ReadChunkBytes (iff, buf, size)
          d0                     a0   a1    d0

        LONG             actual;
        struct IFFHandle *iff;
        UBYTE            *buf;
        LONG             size;

   FUNCTION
        Reads the IFFHandle stream into the buffer for the specified number
        of bytes.  Reads are limited to the size of the current chunk and
        attempts to read past the end of the chunk will truncate.  Function
        returns positive number of bytes read or a negative error code.

   INPUTS
        iff     - pointer to IFFHandle struct.
        buf     - pointer to buffer area to receive data.
        size    - number of bytes to read.

   RESULT
        actual  - (positive) number of bytes read if successful or a
                  (negative) IFFERR_#? error code if not successful.

   EXAMPLE

   NOTES

   BUGS

   SEE ALSO
        ReadChunkRecords(), ParseIFF(), WriteChunkBytes()
```

iffparse.library Page 38

```
iffparse.library/ReadChunkRecords                iffparse.library/ReadChunkRecords

   NAME
        ReadChunkRecords -- Read record elements from the current chunk into
                            a buffer.

   SYNOPSIS
        actual = ReadChunkRecords (iff, buf, recsize, numrec)
          d0                       a0   a1    d0       d1

        LONG             actual;
        struct IFFHandle *iff;
        UBYTE            *buf;
        LONG             recsize, numrec;

   FUNCTION
        Reads records from the current chunk into buffer.  Truncates attempts
        to read past end of chunk (only whole records are read; remaining
        bytes that are not of a whole record size are left unread and
        available for ReadChunkBytes()).

   INPUTS
        iff     - pointer to IFFHandle struct.
        buf     - pointer to buffer area to receive data.
        recsize - size of data records to read.
        numrec  - number of data records to read.

   RESULT
        actual  - (positive) number of whole records read if successful or a
                  (negative) IFFERR_#? error code if not successful.

   EXAMPLE

   NOTES

   BUGS

   SEE ALSO
        ReadChunkBytes(), ParseIFF(), WriteChunkRecords()
```

```
iffparse.library/SetLocalItemPurge        iffparse.library/SetLocalItemPurge

   NAME
       SetLocalItemPurge -- Set purge vector for a local context item.

   SYNOPSIS
       SetLocalItemPurge (item, purgehook)
                          a0     a1

       struct LocalContextItem *item;
       struct Hook             *purgehook;

   FUNCTION
       Sets a local context item to use a client-supplied cleanup (purge)
       vector for disposal when its context is popped.  The purge vector
       will be called when the ContextNode containing this local item is
       popped off the context stack and is about to be deleted itself.  If
       the purge vector has not been set, the parser will use FreeLocalItem
       to delete the item, but if this function is used to set the purge
       vector, the supplied vector will be called with the following
       arguments:

               A0:     pointer to purgehook.
               A2:     pointer to LocalContextItem to be freed.
               A1:     pointer to a LONG containing the value
                       IFFCMD_PURGELCI.

       The user purge vector is then responsible for calling FreeLocalItem()
       as part of its own cleanup.  Although the purge vector can return a
       value, it will be ignored -- purge vectors must always work (best to
       return 0, though).

   INPUTS
       item      - pointer to local context item.
       purgehook - pointer to a Hook structure.

   RESULT

   EXAMPLE

   NOTES

   BUGS

   SEE ALSO
       AllocLocalItem(), FreeLocalItem(), utility/hooks.h
```

```
iffparse.library/StopChunk                          iffparse.library/StopChunk

   NAME
       StopChunk -- Declare a chunk which should cause ParseIFF to return.

   SYNOPSIS
       error = StopChunk (iff, type, id)
        d0                a0    d0   d1

       LONG             error;
       struct IFFHandle *iff;
       LONG             type;
       LONG             id;

   FUNCTION
       Installs an entry handler for the specified chunk which will cause
       the ParseIFF() function to return control to the caller when this
       chunk is encountered.  This is only of value when ParseIFF() is
       called with the IFFPARSE_SCAN control code.

   INPUTS
       iff     - pointer to IFFHandle struct (need not be open).
       type    - type code for chunk to declare (ex. "ILBM").
       id      - identifier for chunk to declare (ex. "BODY").

   RESULT
       error   - 0 if successful or an IFFERR_#? error code if not
                 successful.

   EXAMPLE

   NOTES

   BUGS

   SEE ALSO
       StopChunks(), ParseIFF()
```

iffparse.library Page 41

```
iffparse.library/StopChunks                          iffparse.library/StopChunks

   NAME
       StopChunks -- Declare many stop chunks at once.

   SYNOPSIS
       error = StopChunks (iff, list, n)
        d0                 a0    a1  d0

       LONG            error;
       struct IFFHandle *iff;
       LONG            *list;
       LONG            n;

   FUNCTION
       (is to StopChunk() as PropChunks() is to PropChunk().)

   INPUTS
       iff      - pointer to IFFHandle struct.
       list     - pointer to array of longword chunk types and identifiers.
       n        - number of chunks to declare.

   RESULT
       error    - 0 if successful or an IFFERR_#? error code if not
                  successful.

   EXAMPLE

   NOTES

   BUGS

   SEE ALSO
       StopChunk()
```

iffparse.library Page 42

```
iffparse.library/StopOnExit                          iffparse.library/StopOnExit

   NAME
       StopOnExit -- Declare a stop condition for exiting a chunk.

   SYNOPSIS
       error = StopOnExit (iff, type, id)
        d0                 a0    d0   d1

       LONG            error;
       struct IFFHandle *iff;
       LONG            type;
       LONG            id;

   FUNCTION
       Installs an exit handler for the specified chunk which will cause the
       ParseIFF() function to return control to the caller when this chunk
       is exhausted.  ParseIFF() will return IFFERR_EOC when the declared
       chunk is about to be popped.  This is only of value when ParseIFF()
       is called with the IFFPARSE_SCAN control code.

   INPUTS
       iff      - pointer to IFFHandle struct (need not be open).
       type     - type code for chunk to declare (ex. "ILBM").
       id       - identifier for chunk to declare (ex. "BODY").

   RESULT
       error    - 0 if successful or an IFFERR_#? error code if not
                  successful.

   EXAMPLE

   NOTES

   BUGS

   SEE ALSO
       ParseIFF()
```

```
iffparse.library/StoreItemInContext      iffparse.library/StoreItemInContext

   NAME
        StoreItemInContext -- Store local context item in given context node.

   SYNOPSIS
        StoreItemInContext (iff, item, cn)
                            a0    a1   a2

        struct IFFHandle        *iff;
        struct LocalContextItem *item;
        struct ContextNode      *cn;

   FUNCTION
        Adds the LocalContextItem to the list of items for the given context
        node.  If an LCI with the same Type, ID, and Ident is already
        present in the ContextNode, it will be purged and replaced with the
        new one.  This is a raw form of StoreLocalItem.

   INPUTS
        iff     - pointer to IFFHandle struct for this context.
        item    - pointer to a LocalContextItem to be stored.
        cn      - pointer to context node in which to store item.

   RESULT

   EXAMPLE

   NOTES

   BUGS

   SEE ALSO
        StoreLocalItem()
```

```
iffparse.library/StoreLocalItem              iffparse.library/StoreLocalItem

   NAME
        StoreLocalItem -- Insert a local context item into the context stack.

   SYNOPSIS
        error = StoreLocalItem (iff, item, position)
         d0                     a0    a1     d0

        LONG                    error;
        struct IFFHandle        *iff;
        struct LocalContextItem *item;
        LONG                    position;

   FUNCTION
        Adds the local context item to the list of items for one of the
        context nodes on the context stack and purges any other item in the
        same context with the same ident, type and id.  The position argument
        determines where in the stack to add the item:

        IFFSLI_ROOT:
                Add item to list at root (default) stack position.
        IFFSLI_TOP:
                Add item to the top (current) context node.
        IFFSLI_PROP:
                Add element in top property context.  Top property context is
                either the top FORM chunk, or the top LIST chunk, whichever
                is closer to the top of the stack.

        Items added to the root context, or added to the top context before
        the IFFHandle has been opened or after it has been closed, are put in
        the default context.  That is, they will be the local items found
        only after all other context nodes have been searched.  Items in the
        default context are also immune to being purged until the IFFHandle
        struct itself is deleted with FreeIFF().  This means that handlers
        installed in the root context will still be there after an IFFHandle
        struct has been opened and closed.  (Note that this implies that
        items stored in a higher context will be deleted when that context
        ends.)

   INPUTS
        iff     - pointer to IFFHandle struct.
        item    - pointer to LocalContextItem struct to insert.
        position- where to store the item (IFFSLI_ROOT, _TOP or _PROP).

   RESULT
        error   - 0 if successful or an IFFERR_#? error code if not
                  successful.

   EXAMPLE

   NOTES

   BUGS

   SEE ALSO
        FindLocalItem(), StoreItemInContext(), EntryHandler(), ExitHandler()
```

```
iffparse.library/WriteChunkBytes           iffparse.library/WriteChunkBytes

   NAME
        WriteChunkBytes -- Write data from a buffer into the current chunk.

   SYNOPSIS
        error = WriteChunkBytes (iff, buf, size)
         d0                      a0   a1    d0

        LONG             error;
        struct IFFHandle *iff;
        UBYTE            *buf;
        LONG             size;

   FUNCTION
        Writes "size" bytes from the specified buffer into the current chunk.
        If the current chunk was pushed with IFFSIZE_UNKNOWN, the size of the
        chunk gets increased by the size of the buffer written.  If the size
        was specified for this chunk, attempts to write past the end of the
        chunk will be truncated.

   INPUTS
        iff     - pointer to IFFHandle struct.
        buf     - pointer to buffer area with bytes to be written.
        size    - number of bytes to write.

   RESULT
        error   - (positive) number of bytes written if successful or a
                  (negative) IFFERR_#? error code if not successful.

   EXAMPLE

   NOTES

   BUGS

   SEE ALSO
        PushChunk(), PopChunk(), WriteChunkRecords()
```

```
iffparse.library/WriteChunkRecords         iffparse.library/WriteChunkRecords

   NAME
        WriteChunkRecords -- Write records from a buffer to the current
                             chunk.

   SYNOPSIS
        error = WriteChunkRecords (iff, buf, recsize, numrec)
         d0                        a0   a1    d0       d1

        LONG             error;
        struct IFFHandle *iff;
        UBYTE            *buf;
        LONG             recsize, numrec;

   FUNCTION
        Writes record elements from the buffer into the top chunk.  This
        function operates much like ReadChunkBytes().

   INPUTS
        iff     - pointer to IFFHandle struct.
        buf     - pointer to buffer area containing data.
        recsize - size of data records to write.
        numrec  - number of data records to write.

   RESULT
        error   - (positive) number of whole records written if successful
                  or a (negative) IFFERR_#? error code if not successful.

   EXAMPLE

   NOTES

   BUGS

   SEE ALSO
        WriteChunkBytes()
```

TABLE OF CONTENTS

intuition.library/ActivateGadget
intuition.library/ActivateWindow
intuition.library/AddClass
intuition.library/AddGadget
intuition.library/AddGList
intuition.library/AllocRemember
intuition.library/AutoRequest
intuition.library/BeginRefresh
intuition.library/BuildEasyRequestArgs
intuition.library/BuildSysRequest
intuition.library/ChangeWindowBox
intuition.library/ClearDMRequest
intuition.library/ClearMenuStrip
intuition.library/ClearPointer
intuition.library/CloseScreen
intuition.library/CloseWindow
intuition.library/CloseWorkBench
intuition.library/CurrentTime
intuition.library/DisplayAlert
intuition.library/DisplayBeep
intuition.library/DisposeObject
intuition.library/DoubleClick
intuition.library/DrawBorder
intuition.library/DrawImage
intuition.library/DrawImageState
intuition.library/EasyRequestArgs
intuition.library/EndRefresh
intuition.library/EndRequest
intuition.library/EraseImage
intuition.library/FreeClass
intuition.library/FreeRemember
intuition.library/FreeScreenDrawInfo
intuition.library/FreeSysRequest
intuition.library/GadgetMouse
intuition.library/GetAttr
intuition.library/GetDefaultPubScreen
intuition.library/GetDefPrefs
intuition.library/GetPrefs
intuition.library/GetScreenData
intuition.library/GetScreenDrawInfo
intuition.library/InitRequester
intuition.library/IntuiTextLength
intuition.library/ItemAddress
intuition.library/LockIBase
intuition.library/LockPubScreen
intuition.library/LockPubScreenList
intuition.library/MakeClass
intuition.library/MakeScreen
intuition.library/ModifyIDCMP
intuition.library/ModifyProp
intuition.library/MoveScreen
intuition.library/MoveWindow
intuition.library/MoveWindowInFrontOf
intuition.library/NewModifyProp
intuition.library/NewObject
intuition.library/NextObject
intuition.library/NextPubScreen
intuition.library/ObtainGIRPort
intuition.library/OffGadget
intuition.library/OffMenu
intuition.library/OnGadget
intuition.library/OnMenu
intuition.library/OpenScreen
intuition.library/OpenScreenTagList

intuition.library/OpenWindow
intuition.library/OpenWindowTagList
intuition.library/OpenWorkBench
intuition.library/PointInImage
intuition.library/PrintIText
intuition.library/PubScreenStatus
intuition.library/QueryOverscan
intuition.library/RefreshGadgets
intuition.library/RefreshGList
intuition.library/RefreshWindowFrame
intuition.library/ReleaseGIRPort
intuition.library/RemakeDisplay
intuition.library/RemoveClass
intuition.library/RemoveGadget
intuition.library/RemoveGList
intuition.library/ReportMouse
intuition.library/Request
intuition.library/ResetMenuStrip
intuition.library/RethinkDisplay
intuition.library/ScreenToBack
intuition.library/ScreenToFront
intuition.library/SetAttrsA
intuition.library/SetDefaultPubScreen
intuition.library/SetDMRequest
intuition.library/SetEditHook
intuition.library/SetGadgetAttrsA
intuition.library/SetMenuStrip
intuition.library/SetMouseQueue
intuition.library/SetPointer
intuition.library/SetPrefs
intuition.library/SetPubScreenModes
intuition.library/SetWindowTitles
intuition.library/ShowTitle
intuition.library/SizeWindow
intuition.library/SysReqHandler
intuition.library/UnlockIBase
intuition.library/UnlockPubScreen
intuition.library/UnlockPubScreenList
intuition.library/ViewAddress
intuition.library/ViewPortAddress
intuition.library/WBenchToBack
intuition.library/WBenchToFront
intuition.library/WindowLimits
intuition.library/WindowToBack
intuition.library/WindowToFront
intuition.library/ZipWindow

intuition.library Page 3

```
intuition.library/ActivateGadget               intuition.library/ActivateGadget

   NAME
       ActivateGadget -- Activate a (string or custom) gadget.

   SYNOPSIS
       Success = ActivateGadget( Gadget, Window, Request )
       D0                        A0      A1      A2

       BOOL ActivateGadget( struct Gadget *, struct Window *,
               struct Requester * );

   FUNCTION
       Activates a string or custom gadget.  If successful, this means
       that the user does not need to click in the gadget before typing.

       The window parameter must point to the window which contains the gadget.
       If the gadget is actually in a requester, the window must contain
       the requester, and a pointer to the requester must also be
       passed. The requester parameter must only be valid if the gadget
       has the GTYP_REQGADGET flag set, a requirement for all requester
       gadgets.

       The success of this function depends on a rather complex set
       of conditions.  The intent is that the user is never interrupted from
       what interactions he may have underway.

       The current set of conditions includes:
       -   The window must be active.  If you are opening a new window
           and want an active gadget in it, it is not sufficient to
           assume that the WFLG_ACTIVATE flag has taken effect by the time
           OpenWindow() returns, even if you insert a delay of some
           finite amount of time.  Use the IDCMP_ACTIVEWINDOW IntuiMessage
           to tell when your window really becomes active.  Many
           programs use an event loop that calls ActivateGadget()
           whenever they receive the IDCMP_ACTIVEWINDOW message, and also
           the IDCMP_MOUSEBUTTONS messages, and so on, to keep the
           gadget active until it is used (or the user selects some
           other "Cancel" gadget).

       -   No other gadgets may be in use.  This includes system gadgets,
           such as those for window sizing, dragging, etc.
       -   If the gadget is in a requester, that requester must
           be active. (Use IDCMP_REQSET and IDCMP_REQCLEAR).
       -   The right mouse button cannot be held down (e.g. menus)

       NOTE: Don't try to activate a gadget which is disabled or
       not attached to a window or requester.

   INPUTS
       Gadget = pointer to the gadget that you want activated.
       Window = pointer to a window structure containing the gadget.
       Requester = pointer to a requester (may by NULL if this isn't
          a requester gadget (i.e. GTYP_REQGADGET is not set)).

   RESULT
       If the conditions above are met, and the gadget is in fact a string
       gadget, then this function will return TRUE, else FALSE.

   BUGS
       At present, this function will not return FALSE if a custom
       gadget declines to be activated.

   SEE ALSO
```

intuition.library Page 4

```
intuition.library/ActivateWindow               intuition.library/ActivateWindow

   NAME
       ActivateWindow -- Activate an Intuition window.

   SYNOPSIS
       [success =] ActivateWindow( Window )
       [D0]                        A0

       [LONG] ActivateWindow( struct Window * );
       /* returns LONG in V36 and higher */

   FUNCTION
       Activates an Intuition window.

       Note that this call may have its action deferred: you cannot assume
       that when this call is made the selected window has become active.
       This action will be postponed while the user plays with gadgets and
       menus, or sizes and drags windows.  You may detect when the window
       actually has become active by the IDCMP_ACTIVEWINDOW IDCMP message.

       This call is intended to provide flexibility but not to confuse the
       user.  Please call this function synchronously with some action
       by the user.

   INPUTS
       Window = a pointer to a Window structure

   RESULT
       V35 and before: None.
       V36 and later: returns zero if no problem queuing up
           the request for deferred action

   BUGS
       Calling this function in a tight loop can blow out Intuition's deferred
       action queue.

   SEE ALSO
       OpenWindow(), and the WFLG_ACTIVATE window flag
```

```
intuition.library/AddClass                            intuition.library/AddClass

    NAME
        AddClass -- Make a public class available (V36)

    SYNOPSIS
        AddClass( Class )
                  A0

        VOID AddClass( struct IClass * );

    FUNCTION
        Adds a public boopsi class to the internal list of classes available
        for public consumption.

        You must call this function after you call MakeClass().

    INPUTS
        Class = pointer returned by MakeClass()

    RESULT
        Nothing returned.

    NOTES

    BUGS
        Although there is some protection against creating classes
        with the same name as an existing class, this function
        does not do any checking or other dealings with like-named
        classes.  Until this is rectified, only officially registered
        names can be used for public classes, and there is no
        "class replacement" policy in effect.

    SEE ALSO
        MakeClass(), FreeClass(), RemoveClass()
        Document "Basic Object-Oriented Programming System for Intuition"
        and the "boopsi Class Reference" document.
```

```
intuition.library/AddGadget                          intuition.library/AddGadget

    NAME
        AddGadget -- Add a gadget to the gadget list of a window.

    SYNOPSIS
        RealPosition = AddGadget( Window, Gadget, Position )
        D0                        A0      A1      D0

        UWORD  AddGadget( struct Window *, struct Gadget *, UWORD );

    FUNCTION
        Adds the specified gadget to the gadget list of the given window,
        linked in at the position in the list specified by the position
        argument (that is, if Position == 0, the gadget will be inserted
        at the head of the list, and if Position == 1 then the gadget will
        be inserted after the first gadget and before the second).  If the
        position you specify is greater than the number of gadgets in the
        list, your gadget will be added to the end of the list.

        Calling AddGadget() does not cause your gadget to be redisplayed.
        The benefit of this is that you may add several gadgets without
        having the gadget list redrawn every time.

        This procedure returns the position at which your gadget was added.

        NOTE:  A relatively safe way to add the gadget to the end of the
        list is to specify a position of -1 (i.e., (UWORD) ~0).  That way,
        only the 65536th (and multiples of it) will be inserted at the wrong
        position.  The return value of the procedure will tell you where it was
        actually inserted.

        NOTE:  The system window gadgets are initially added to the
        front of the gadget list.  The reason for this is:  If you position
        your own gadgets in some way that interferes with the graphical
        representation of the system gadgets, the system's ones will be "hit"
        first by user.  If you then start adding gadgets to the front of the
        list, you will disturb this plan, so beware.  On the other hand, if
        you don't violate the design rule of never overlapping your gadgets,
        there's no problem.

        NOTE:  You may not add your own gadgets to a screen.  Gadgets may
        be added to backdrop windows, however, which can be visually similar,
        but also provide an IDCMP channel for gadget input messages.

    INPUTS
        Window = pointer to the window to get your gadget
        Gadget = pointer to the new gadget
        Position = integer position in the list for the new gadget (starting
            from zero as the first position in the list)

    RESULT
        Returns the position of where the gadget was actually added.

    BUGS

    SEE ALSO
        AddGList(), RemoveGadget(), RemoveGList()
```

intuition.library Page 7

```
intuition.library/AddGList                              intuition.library/AddGList

    NAME
        AddGList -- Add a linked list of gadgets to a window or requester.

    SYNOPSIS
        RealPosition = AddGList( Window, Gadget, Position, Numgad, Requester )
        D0                       A0      A1      D0        D1      A2

        UWORD AddGList( struct Window *, struct Gadget *, UWORD, WORD,
                struct Requester * );

    FUNCTION
        Adds the list of gadgets to the gadget list of the given window
        or requester linked in at the position in the list specified by
        the position argument.

        See AddGadget() for more information about gadget list position,
        and more information about gadgets in general.

        The requester parameter will be ignored unless the GTYP_REQGADGET
        bit is set in the GadgetType field of the first gadget in the list.
        In that case, the gadget list is added to the requester gadgets.
        NOTE: be sure that GTYP_REQGADGET is either set or cleared consistently
        for all gadgets in the list.  NOTE ALSO: The window parameter
        should point to the window that the requester (will) appear in.

        Will add 'Numgad' gadgets from gadget list linked by the field
        NextGadget, or until some NextGadget field is found to be NULL.  Does
        not assume that the Numgad'th gadget has NextGadget equal to NULL.

        NOTE WELL: In order to link your gadget list in, the NextGadget
        field of the Numgad'th (or last) gadget will be modified.  Thus, if
        you are adding the first 3 gadgets from a linked list of five gadgets,
        this call will sever the connection between your third and fourth
        gadgets.

    INPUTS
        Window = pointer to the window to get your gadget
        Gadget = pointer to the first gadget to be added
        Position = integer position in the list for the new gadget
            (starting from zero as the first position in the list)
        Numgad = the number of gadgets from the linked list to be added
            if Numgad equals -1, the entire null-terminated list of
            gadgets will be added.
        Requester = the requester the gadgets will be added to if the
            GTYP_REQGADGET GadgetType flag is set for the first gadget
            in the list

    RESULT
        Returns the position of where the first gadget in the list was actually
        added.

    BUGS

    SEE ALSO
        AddGadget(), RemoveGadget(), RemoveGList()
```

intuition.library Page 8

```
intuition.library/AllocRemember                    intuition.library/AllocRemember

    NAME
        AllocRemember -- AllocMem() with tracking to make freeing easy.

    SYNOPSIS
        MemBlock = AllocRemember( RememberKey, Size, Flags )
        D0                        A0           D0    D1

        APTR AllocRemember( struct Remember **, ULONG, ULONG );

    FUNCTION
        This routine calls the Exec AllocMem() function for you, but also links
        the parameters of the allocation into a master list, so that
        you can simply call the Intuition routine FreeRemember() at a later
        time to deallocate all allocated memory without being required to
        remember the details of the memory you've allocated.

        This routine will have two primary uses:
      -    Let's say that you're doing a long series of allocations in a
           procedure.  If any one of the allocations fails, your program
            may need to abort the procedure.  Abandoning ship correctly
            involves freeing up what memory you've already allocated.  This
            procedure allows you to free up that memory easily, without being
            required to keep track of how many allocations you've already done,
            what the sizes of the allocations were, or where the memory was
            allocated.

      -    Also, in the more general case, you may do all of the allocations
           in your entire program using this routine.  Then, when your
           program is exiting, you can free it all up at once with a
           simple call to FreeRemember().

        You create the "anchor" for the allocation master list by creating
        a variable that's a pointer to struct Remember, and initializing
        that pointer to NULL.  This is called the RememberKey.  Whenever
        you call AllocRemember(), the routine actually does two memory
        allocations, one for the memory you want and the other for a copy
        of a Remember structure.  The Remember structure is filled in
        with data describing your memory allocation, and it's linked
        into the master list pointed to by your RememberKey.  Then, to
        free up any memory that's been allocated, all you have to do is
        call FreeRemember() with your RememberKey.

        Please read the FreeRemember() function description, too.  As you will
        see, you can select either to free just the link nodes and keep all the
        allocated memory for yourself, or to free both the nodes and your memory
        buffers.


    INPUTS
        RememberKey = the address of a pointer to struct Remember.  Before the
        very first call to AllocRemember, initialize this pointer to NULL.

        Size = the size in bytes of the memory allocation.  Please refer to the
            exec.library/AllocMem() function for details.
        Flags = the specifications for the memory allocation.  Please refer to
            the exec.library/AllocMem() function for details.

    EXAMPLE
        struct Remember *RememberKey;
        RememberKey = NULL;
        buffer = AllocRemember(&RememberKey, BUFSIZE, MEMF_CHIP);
        if (buffer)
        {
            /* Use the buffer */
        ...
    }
    FreeRemember(&RememberKey, TRUE);

RESULT
    If the memory allocation is successful, this routine returns the byte
    address of your requested memory block.  Also, the node to your block
    will be linked into the list pointed to by your RememberKey variable.
    If the allocation fails, this routine returns NULL and the list pointed
    to by RememberKey, if any, will be unchanged.

BUGS
    This function makes two allocations for each memory buffer
    you request.  This is neither fast nor good for memory
    fragmentation.

    This function should use the exec AllocPool() function internally,
    at least for the Remember headers.

SEE ALSO
    FreeRemember(), exec.library/AllocMem()
```

```
intuition.library/AutoRequest                         intuition.library/AutoRequest

   NAME
       AutoRequest -- Automatically build and get response from a requester.

   SYNOPSIS
       Response = AutoRequest( Window, BodyText, PosText, NegText,
       D0                      A0      A1        A2       A3
                       PosFlags, NegFlags, Width, Height )
                       D0        D1        D2     D3

       BOOL AutoRequest( struct Window *, struct IntuiText *,
               struct IntuiText *, struct IntuiText *, WORD, WORD );

   FUNCTION
       This procedure automatically builds a requester for you and then
       waits for a response from the user, or for the system to satisfy your
       request.  If the response is positive, this procedure returns TRUE.
       If the response is negative, this procedure returns FALSE.

       An IDCMPFlag specification is created by bitwise "or'ing" your
       PosFlags, NegFlags, and the IDCMP classes IDCMP_GADGETUP and
       IDCMP_RAWKEY.  You may specify zero flags for either the PosFlags
       or NegFlags arguments.

       The IntuiText arguments, and the width and height values, are
       passed directly to the BuildSysRequest() procedure along with
       your window pointer and the IDCMP flags.  Please refer to
       BuildSysRequest() for a description of the IntuiText that you are
       expected to supply when calling this routine.  It's an important
       but long-winded description that need not be duplicated here.

       If the BuildSysRequest() procedure does not return a pointer
       to a window, it will return TRUE or FALSE (not valid structure
       pointers) instead, and these BOOL values will be returned to
       you immediately.

       On the other hand, if a valid window pointer is returned, that
       window will have had its IDCMP ports and flags initialized according
       to your specifications.  AutoRequest() then waits for IDCMP messages
       on the UserPort, which satisfies one of four requirements:
       -   either the message is of a class that matches
           one of your PosFlags arguments (if you've supplied
           any), in which case this routine returns TRUE.  Or
       -   the message class matches one of your NegFlags
           arguments (if you've supplied any), in which case
           this routine returns FALSE.  Or
       -   the IDCMP message is of class IDCMP_GADGETUP, which means that
           one of the two gadgets, as provided with the PosText and NegText
           arguments, was selected by the user.  If the TRUE gadget
           was selected, TRUE is returned.  If the FALSE gadget was
           selected, FALSE is returned.
       -   Lastly, two IDCMP_RAWKEY messages may satisfy the request: those
           for the V and B keys with the left Amiga key depressed.
           These keys, satisfy the gadgets on the left or right side of
           the requester--TRUE or FALSE--, respectively.

           NOTE: For V36, these two keys left-Amiga-B and V are processed
           through the default keymap.

       When the dust has settled, this routine calls FreeSysRequest() if
       necessary to clean up the requester and any other allocated memory.

       NOTE: For V36, this function now switches the processor stack
       to ensure sufficient stack space for the function to succeed.
```

```
INPUTS
    Window = pointer to a Window structure.  See BuildSysRequest()
        for a full discussion.
    BodyText = pointer to an IntuiText structure
    PosText = pointer to an IntuiText structure, may by NULL.
    NegText = pointer to an IntuiText structure, MUST be valid!
    PosFlags = flags for the IDCMP
    NegFlags = flags for the IDCMP
    Width, Height = the sizes to be used for the rendering of the requester

    NOTE for V36: The width and height parameters are ignored, as
    are several other specifications in the IntuiText, to make
    AutoRequest() requesters retroactively conform to the new look
    designed for EasyRequest().

RESULT
    The return value is either TRUE or FALSE.  See the text above for a
    complete description of the chain of events that might lead to either
    of these values being returned.

BUGS
    The requester no longer devolves into a call to DisplayAlert()
    if there is not enough memory for the requester.

SEE ALSO
    EasyRequest(), BuildSysRequest(), SysReqHandler()
```

```
intuition.library/BeginRefresh                    intuition.library/BeginRefresh

    NAME
        BeginRefresh -- Sets up a window for optimized refreshing.

    SYNOPSIS
        BeginRefresh( Window )
                      A0

        VOID BeginRefresh( struct Window * );

    FUNCTION
        This routine sets up your window for optimized refreshing.

        Its role is to provide Intuition integrated access to the Layers
        library function BeginUpdate().  Its additional contribution is
        to be sure that locking protocols for layers are followed, by
        locking both layers of a WFLG_GIMMEZEROZERO window only after the
        parent Layer_Info has been locked.  Also, the WFLG_WINDOWREFRESH
        flag is set in your window, for your information.

        The purpose of BeginUpdate(), and hence BeginRefresh(), is to
        restrict rendering in a window (layer) to the region that needs
        refreshing after an operation such as window sizing or uncovering.
        This restriction to the "damage region" persists until you call
        EndRefresh().

        For instance, if you have a WFLG_SIMPLE_REFRESH window which is
        partially concealed and the user brings it to the front, you can
        receive an IDCMP_REFRESHWINDOW message asking you to refresh your
        display.  If you call BeginRefresh() before doing any of the
        rendering, then the layer that underlies your window will be arranged
        so that the only rendering that will actually take place will be that
        which goes to the newly-revealed areas.  This is very performance-
        efficient, and visually attractive.

        After you have performed your refresh of the display, you should call
        EndRefresh() to reset the state of the layer and the window.  Then you
        may proceed with rendering to the entire window as usual.

        You learn that your window needs refreshing by receiving either a
        message of class IDCMP_REFRESHWINDOW through the IDCMP, or an input
        event of class IECLASS_REFRESHWINDOW through the Console device.
        Whenever you are told that your window needs refreshing, you should
        call BeginRefresh() and EndRefresh() to clear the refresh-needed
        state, even if you don't plan on doing any rendering.  You may relieve
        yourself of even this burden by setting the WFLG_NOCAREREFRESH flag
        when opening your window.

        WARNING: You should only perform graphics refreshing operations
        during the period between calling BeginRefresh() and EndRefresh().
        In particular, do not call RefreshGadgets() or RefreshGList(), since
        the locking protocol internal to Intuition runs the risk of creating
        a deadlock.  Note that Intuition refreshes the gadgets (through
        the damage region) before it sends the IDCMP_REFRESHWINDOW message.

        ANOTHER WARNING: The concept of multiple refresh passes using
        EndRefresh( w, FALSE ) is not completely sound without further
        protection.  The reason is that between two sessions, more
        damage can occur to your window.  Your final EndRefresh( w, TRUE )
        will dispose of all damage, including the new, and your
        initial refreshing pass will never get the chance to refresh
        the new damage.

        To avoid this, you must protect your session using LockLayerInfo()
        which will prevent Intuition from performing window operations
        or anything else which might cause further damage from occurring.
        Again, while holding the LayerInfo lock make no Intuition
        function calls dealing with gadgets; just render.

        You can, however, call InstallClipRegion() for the different
        refresh passes, if you have two clip regions.

        SIMILAR WARNING: Your program and Intuition "share" your window
        layer's DamageList.  BeginRefresh() helps arbitrate this
        sharing, but the lower-level function layers.library/BeginUpdate()
        does not.  It isn't really supported to use BeginUpdate() on
        a window's layer, but if you do--for whatever reason--it is
        critical that you first acquire the LayerInfo lock as in
        the above example: even if you only have one pass of refresh
        rendering to do.  Otherwise, the refreshing of your window's
        borders and gadgets can be incomplete, and the problem might
        occur only under certain conditions of task priority and
        system load.

    EXAMPLE
        Code fragment for "two pass" window refreshing, in response
        to an IDCMP_REFRESHWINDOW message:
        switch ( imsg->Class )
        {
        ...
        case IDCMP_REFRESHWINDOW:
            window = imsg->IDCMPWindow;

            /* this lock only needed for "two-pass" refreshing */
            LockLayerInfo( &window->WScreen->LayerInfo );

            /* refresh pass for region 1 */
            origclip = InstallClipRegion( window->WLayer, region1 );
            BeginRefresh( window );
            myRefreshRegion1( window );
            EndRefresh( window, FALSE );

            /* refresh pass for region 2 */
            InstallClipRegion( window->WLayer, region2 );
            BeginRefresh( window );
            myRefreshRegion2( window );
            EndRefresh( window, TRUE );         /* and dispose damage list */

            /* restore and unlock */
            InstallClipRegion( window->WLayer, origclip );
            UnlockLayerInfo( &window->WScreen->LayerInfo );
            break;
        ...
        }


    INPUTS
        Window = pointer to the window structure which needs refreshing

    RESULT
        None

    BUGS
        This function should check the return code of
        layers.library/BeginUpdate(), and abort if that function fails.

    SEE ALSO
       EndRefresh(), layers.library/BeginUpdate(), OpenWindow()
        layer.library/InstallClipRegion(), graphics.library/LockLayerInfo()
        The "Windows" chapter of the Intuition Reference Manual
```

```
intuition.library/BuildEasyRequestArgs intuition.library/BuildEasyRequestArgs

    NAME
        BuildEasyRequestArgs -- Simple creation of system request. (V36)
        BuildEasyRequest -- Varargs stub for BuildEasyRequestArgs(). (V36)

    SYNOPSIS
        ReqWindow = BuildEasyRequestArgs( RefWindow, easyStruct, IDCMP, Args )
        D0                                A0         A1          D0     A3

        struct Window *BuildEasyRequestArgs( struct Window *,
                struct EasyStruct *, ULONG, APTR );

        ReqWindow = BuildEasyRequest( RefWindow, easyStruct, IDCMP, Arg1, ... )

        struct Window *BuildEasyRequest( struct Window *,
                struct EasyStruct *, ULONG, APTR, ... );

    FUNCTION
        This function is to EasyRequest() as BuildSysRequest() is to
        AutoRequest(): it returns a pointer to the system requester
        window.  The input from that window can then be processed
        under application control.

        It is recommended that this processing be done with
        SysReqHandler(), so that future enhancement to the
        processing will be enjoyed.

        After you have determined that the requester is satisfied or
        cancelled, you must free this requester using FreeSysRequest().

        Please see the autodoc for EasyRequest().

        NOTE: This function switches the processor stack to ensure
        sufficient stack space for the function to complete.

    INPUTS
        Window = reference window for requester: determines the
            requester window title and screen.
        easyStruct = pointer to EasyStruct structure, as described
            in the EasyRequest() autodocs.
        IDCMP = (NOT A POINTER) provided application specific IDCMP
            flags for the system requester window.
        Args = see EasyRequest()

    RESULT
        A pointer to the system request window opened.  In the event
        of problems, you may also be returned the value '0' which
        is to be interpreted as the "FALSE, Cancel" choice, or
        (if you have a second gadget defined) the value '1', which
        is to be taken to mean the equivalent of your corresponding
        left-most gadget.

        If there is a problem creating the window, a recoverable alert may
        be substituted for the requester, and the result, either 0 or 1,
        returned.

    BUGS
        Does not put up alternative alert.
        See also BUGS listed for EasyRequestArgs().

    SEE ALSO
        EasyRequestArgs(), FreeSysRequest(), SysReqHandler(),
        BuildSysRequest(), AutoRequest()
```

```
intuition.library/BuildSysRequest             intuition.library/BuildSysRequest

    NAME
        BuildSysRequest -- Build and display a system requester.

    SYNOPSIS
        ReqWindow = BuildSysRequest( Window, BodyText, PosText, NegText,
        D0                           A0      A1        A2       A3
                          IDCMPFlags, Width, Height )
                          D0          D2     D3

        struct Window *BuildSysRequest( struct Window *, struct IntuiText *,
                struct IntuiText *, struct IntuiText *, ULONG, WORD, WORD );

    FUNCTION
        This procedure builds a system requester based on the supplied
        information.  If all goes well and the requester is constructed,
        this procedure returns a pointer to the window in which the requester
        appears.  That window will have its IDCMP initialized to reflect the
        flags found in the IDCMPFlags argument.  You may then wait on those
        ports to detect the user's response to your requester, which response
        may include either selecting one of the gadgets or causing some other
        event to be noticed by Intuition (like IDCMP_DISKINSERTED, for
        instance).  After the requester is satisfied, you should call the
        FreeSysRequest() procedure to remove the requester and free up
        any allocated memory.

        See the autodoc for SysReqHandler() for more information on the
        how to handle the IntuiMessages this window will receive.

        The requester used by this function has the NOISYREQ flag bit set,
        which means that the set of IDCMPFlags that may be used here
        include IDCMP_RAWKEY, IDCMP_MOUSEBUTTONS, and others.

        In release previous to V36, if the requester could not be built,
        this function would try to call DisplayAlert() with the same
        information, with more or less favorable results.  In V36,
        the requesters themselves require less memory (SIMPLEREQ), but
        there is no alert attempt.

        The function may return TRUE (1) or FALSE if it cannot post
        the requester.  (V36 will always return FALSE, but be sure to
        test for TRUE in case somebody reinstates the fallback alert.)

        If the window argument you supply is equal to NULL, a new window will
        be created for you in the Workbench screen, or the default
        public screen, for V36.  If you want the requester
        created by this routine to be bound to a particular window (i.e.,
        to appear in the same screen as the window), you should
        not supply a window argument of NULL.

        New for V36: if you pass a NULL window pointer, the system requester
        will appear on the default public screen, which is not always
        the Workbench.

        The text arguments are used to construct the display.  Each is a
        pointer to an instance of the structure IntuiText.

        The BodyText argument should be used to describe the nature of
        the requester.  As usual with IntuiText data, you may link several
        lines of text together, and the text may be placed in various
        locations in the requester.  This IntuiText pointer will be stored
        in the ReqText variable of the new requester.

        The PosText argument describes the text that you want associated
        with the user choice of "Yes,  TRUE,  Retry,  Good."  If the requester
```

```
        is successfully opened, this text will be rendered in a gadget in
        the lower-left of the requester, which gadget will have the
        GadgetID field set to TRUE.  If the requester cannot be opened and
        the DisplayAlert() mechanism is used, this text will be rendered in
        the lower-left corner of the alert display with additional text
        specifying that the left mouse button will select this choice.  This
        pointer can be set to NULL, which specifies that there is no TRUE
        choice that can be made.

        The NegText argument describes the text that you want associated
        with the user choice of "No,  FALSE,  Cancel,  Bad."  If the requester
        is successfully opened, this text will be rendered in a gadget in
        the lower-right of the requester, which gadget will have the
        GadgetID field set to FALSE.  If the requester cannot be opened and
        the DisplayAlert() mechanism is used, this text will be rendered in
        the lower-right corner of the alert display with additional text
        specifying that the right mouse button will select this choice.  This
        pointer cannot be set to NULL.  There must always be a way for the
        user to cancel this requester.

        The Positive and Negative Gadgets created by this routine have
        the following features:
          - GTYP_BOOLGADGET
          - GACT_RELVERIFY
          - GTYP_REQGADGET
          - GACT_TOGGLESELECT

        When defining the text for your gadgets, you may find it convenient
        to use the special constants used by Intuition for the construction
        of the gadgets.  These include defines like AUTODRAWMODE, AUTOLEFTEDGE,
        AUTOTOPEDGE and AUTOFRONTPEN.  You can find these in your local
        intuition.h (or intuition.i) file.

        These hard-coded constants are not very resolution or font
        sensitive, but V36 will override them to provide more modern
        layout.

        New for V36, linked lists of IntuiText are not correctly supported
        for gadget labels.

        The width and height values describe the size of the requester.  All
        of your BodyText must fit within the width and height of your
        requester.  The gadgets will be created to conform to your sizes.

        VERY IMPORTANT NOTE:  for this release of this procedure, a new window
        is opened in the same screen as the one containing your window.
        Future alternatives may be provided as a function distinct from this
        one.

        NOTE: This function will pop the screen the requester and its
        window appears in to the front of all screens.  New for V36,
        if the user doesn't perform any other screen arrangement before
        finishing with the requester, a popped screen will be pushed
        back behind.

    INPUTS
        Window = pointer to a Window structure
        BodyText = pointer to an IntuiText structure
        PosText = pointer to an IntuiText structure
        NegText = pointer to an IntuiText structure
        IDCMPFlags = the IDCMP flags you want used for the initialization of the
           IDCMP of the window containing this requester
        Width, Height = the size required to render your requester

        NOTE for V36: the width and height you pass are ignored, as
        are some of the parameters of your IntuiText, so that Intuition
```

can make the Requesters real nice for the new look.

RESULT

If the requester was successfully created, the value returned by this procedure is a pointer to the window in which the requester is rendered. If the requester could not be created, this routine might have called DisplayAlert() before returning (it depends on the version) and will pass back TRUE if the user pressed the left mouse button and FALSE if the user pressed the right mouse button. If the version of Intuition doesn't call DisplayAlert(), or if it does, and there's not enough memory for the alert, the value of FALSE is returned.

BUGS

This procedure currently opens a window in the Screen which contains the window which is passed as a parameter, or the default public screen, if that parameter is NULL. Although not as originally envisioned, this will probably always be the behavior of this function.

DisplayAlert() is not called in version V36.

It's almost impossible to make complete, correct account of different system fonts, window border dimensions, and screen resolution to get the layout of a System Requester just right using this routine. For V36, we recommend the automatic layout implemented in BuildEasyRequest and EasyRequest.

SEE ALSO

FreeSysRequest(), DisplayAlert(), ModifyIDCMP(), exec.library/Wait(), Request(), AutoRequest(), EasyRequest(), BuildEasyRequestArgs()

intuition.library/ChangeWindowBox   intuition.library/ChangeWindowBox

NAME

ChangeWindowBox -- Change window position and dimensions. (V36)

SYNOPSIS

```
ChangeWindowBox( Window, Left, Top, Width, Height )
                 A0      D0    D1   D2     D3

VOID ChangeWindowBox( struct Window *, WORD, WORD, WORD, WORD );
```

FUNCTION

Makes simultaneous changes in window position and dimensions, in absolute (not relative) coordinates.

Like MoveWindow() and SizeWindow(), the effect of this function is deferred until the next input comes along. Unlike these functions, ChangeWindowBox() specifies absolute window position and dimensions, not relative. This makes for more reliable results considering that the action is deferred, so this function is typically preferable to MoveWindow() and SizeWindow() paired.

You can detect that this operation has completed by receiving the IDCMP_CHANGEWINDOW IDCMP message

The dimensions are limited to legal range, but you should still take care to specify sensible inputs based on the window's dimension limits and the size of its screen.

This function limits the position and dimensions to legal values.

INPUTS

Window = the window to change position/dimension
Left, Top, Width, Height = new position and dimensions

RESULT

Position and dimension are changed to your specification, or as close as possible.
Returns nothing.

BUGS

SEE ALSO

MoveWindow(), SizeWindow(), ZipWindow(), layers.library/MoveSizeLayer()

```
intuition.library/ClearDMRequest              intuition.library/ClearDMRequest

    NAME
        ClearDMRequest -- Clear (detaches) the DMRequest of the window.

    SYNOPSIS
        Response = ClearDMRequest( Window )
        D0                         A0

        BOOL ClearDMRequest( struct Window * );

    FUNCTION
        Attempts to clear the DMRequest from the specified window,
        that is detaches the special requester that you attach to
        the double-click of the menu button which the user can then
        bring up on demand.  This routine WILL NOT clear the DMRequest
        if it's active (in use by the user). The IDCMP message class
        IDCMP_REQCLEAR can be used to detect that the requester is not in use,
        but that message is sent only when the last of perhaps several
        requesters in use in a window is terminated.

    INPUTS
        Window = pointer to the window from which the DMRequest is to be cleared.

    RESULT
        If the DMRequest was not currently in use, detaches the DMRequest
        from the window and returns TRUE.

        If the DMRequest was currently in use, doesn't change anything
        and returns FALSE.

    BUGS

    SEE ALSO
        SetDMRequest(), Request()
```

```
intuition.library/ClearMenuStrip              intuition.library/ClearMenuStrip

    NAME
        ClearMenuStrip -- Clear (detach) the menu strip from the window.

    SYNOPSIS
        ClearMenuStrip( Window )
                        A0

        VOID ClearMenuStrip( struct Window * );

    FUNCTION
        Detaches the current menu strip from the window; menu strips
        are attached to windows using the SetMenuStrip() function
        (or, for V36, ResetMenuStrip() ).

        If the menu is in use (for that matter if any menu is in use)
        this function will block (Wait()) until the user has finished.

        Call this function before you make any changes to the data
        in a Menu or MenuItem structure which is part of a menu
        strip linked into a window.

    INPUTS
        Window = pointer to a window structure

    RESULT
        None

    BUGS

    SEE ALSO
        SetMenuStrip(), ResetMenuStrip()
```

```
intuition.library/ClearPointer                 intuition.library/ClearPointer

    NAME
        ClearPointer -- Clear the mouse pointer definition from a window.

    SYNOPSIS
        ClearPointer( Window )
                      A0

        VOID ClearPointer( struct Window * );

    FUNCTION
        Clears the window of its own definition of the Intuition mouse pointer.
        After calling ClearPointer(), every time this window is the active
        one the default Intuition pointer will be the pointer displayed
        to the user.  If your window is the active one when this routine
        is called, the change will take place immediately.

        Custom definitions of the mouse pointer which this function clears
        are installed by a call to SetPointer().

    INPUTS
        Window = pointer to the window to be cleared of its pointer definition

    RESULT
        None

    BUGS

    SEE ALSO
        SetPointer()
```

```
intuition.library/CloseScreen                   intuition.library/CloseScreen

    NAME
        CloseScreen -- Close an Intuition screen.

    SYNOPSIS
        [Success =] CloseScreen( Screen )
        [D0]                     A0

        [BOOL] CloseScreen( struct Screen * );
        /* returns BOOL in V36 and greater */

    FUNCTION
        Unlinks the screen, unlinks the viewport, deallocates everything that
        Intuition allocated when the screen was opened (using OpenScreen()).
        Doesn't care whether or not there are still any windows attached to the
        screen.  Doesn't try to close any attached windows; in fact, ignores
        them altogether (but see below for changes in V36).

        If this is the last screen to go, attempts to reopen Workbench.

        New for V36: this function will refuse to close the screen
        if there are windows open on the screen when CloseScreen() is
        called.  This avoids the almost certain crash when a screen
        is closed out from under a window.

    INPUTS
        Screen = pointer to the screen to be closed.

    RESULT
        New for V36: returns TRUE (1) if screen is closed,
        returns FALSE (0) if screen had open windows when
        called.

    BUGS

    SEE ALSO
        OpenScreen()
```

```
intuition.library/CloseWindow                          intuition.library/CloseWindow

    NAME
        CloseWindow -- Close an Intuition window.

    SYNOPSIS
        CloseWindow( Window )
                     A0

        VOID CloseWindow( struct Window * );

    FUNCTION
        Closes an Intuition window.  Unlinks it from the system, deallocates
        its memory, and makes it disappear.

        When this function is called, all IDCMP messages which have been sent
        to your window are deallocated.  If the window had shared a message
        Port with other windows, you must be sure that there are no unreplied
        messages for this window in the message queue.  Otherwise, your program
        will try to make use of a linked list (the queue) which contains free
        memory (the old messages).  This will give you big problems.
        See the code fragment CloseWindowSafely(), below.

        NOTE:  If you have added a Menu strip to this Window (via
        a call to SetMenuStrip()) you must be sure to remove that Menu strip
        (via a call to ClearMenuStrip()) before closing your Window.

        NOTE: This function may block until it is safe to de-link and free
        your window.  Your program may thus be suspended while the user
        plays with gadgets, menus, or window sizes and position.

        New for V36: If your window is a "Visitor Window" (see OpenWindow)
        CloseWindow will decrement the "visitor count" in the public screen
        on which the window was open.  When the last visitor window is
        closed, a signal will be sent to the public screen task, if this
        was pre-arranged (see OpenScreen).

    INPUTS
        Window = a pointer to a Window structure

    RESULT
        None

    BUGS

    SEE ALSO
        OpenWindow(), OpenScreen(), CloseScreen()

    EXAMPLE
    /*  CloseWindowSafely */

        /* these functions close an Intuition window
         * that shares a port with other Intuition
         * windows or IPC customers.
         *
         * We are careful to set the UserPort to
         * null before closing, and to free
         * any messages that it might have been
         * sent.
         */
        #include "exec/types.h"
        #include "exec/nodes.h"
        #include "exec/lists.h"
        #include "exec/ports.h"
        #include "intuition/intuition.h"
```

```
        CloseWindowSafely( win )
        struct Window *win;
        {
            /* we forbid here to keep out of race conditions with Intuition */
            Forbid();

            /* send back any messages for this window
             * that have not yet been processed
             */
            StripIntuiMessages( win->UserPort, win );

            /* clear UserPort so Intuition will not free it */
            win->UserPort = NULL;

            /* tell Intuition to stop sending more messages */
            ModifyIDCMP( win, 0L );

            /* turn multitasking back on */
            Permit();

            /* and really close the window */
            CloseWindow( win );
        }

        /* remove and reply all IntuiMessages on a port that
         * have been sent to a particular window
         * (note that we don't rely on the ln_Succ pointer
         *  of a message after we have replied it)
         */
        StripIntuiMessages( mp, win )
        struct MsgPort *mp;
        struct Window *win;
        {
            struct IntuiMessage *msg;
            struct Node *succ;

            msg = (struct IntuiMessage *) mp->mp_MsgList.lh_Head;

            while( succ =  msg->ExecMessage.mn_Node.ln_Succ ) {

                if( msg->IDCMPWindow ==  win ) {

                    /* Intuition is about to free this message.
                     * Make sure that we have politely sent it back.
                     */
                    Remove( msg );

                    ReplyMsg( msg );
                }

                msg = (struct IntuiMessage *) succ;
            }
        }
```

```
intuition.library/CloseWorkBench               intuition.library/CloseWorkBench

   NAME
       CloseWorkBench -- Closes the Workbench screen.

   SYNOPSIS
       Success = CloseWorkBench()
       D0

       LONG CloseWorkBench( VOID );

   FUNCTION
       This routine attempts to close the Workbench screen:
       -   Test whether or not any applications have opened windows on the
           Workbench, and return FALSE if so.  Otherwise ...
       -   Clean up all special buffers
       -   Close the Workbench screen
       -   Make the Workbench program mostly inactive (it will still
           monitor disk activity)
       -   Return TRUE

   INPUTS
       None

   RESULT
       TRUE if the Workbench screen closed successfully
       FALSE if the Workbench was not open, or if it has windows
           open which are not Workbench drawers.

  NOTES
       This routine has been drastically rewritten for V36.
       It is much more solid, although we haven't eliminated
       all the problem cases yet.

   BUGS
       The name of this function is improperly spelled.  Should
       be CloseWorkbench().

       It might be more convenient to have it return TRUE if the
       Workbench wasn't opened when called.  The idea as it is now
       is probably this: if you want to free up the memory of the
       Workbench screen when your program begins, you can call
       CloseWorkBench().  The return value of that call indicates
       whether you should call OpenWorkBench() when your program
       exits: if FALSE, that means either the the Workbench existed
       but you could not close it, or that it wasn't around to
       begin with, and you should not try to re-open it.

       We would prefer that you provide a user selection to attempt
       to open or close the Workbench screen from within your application,
       rather than your making assumptions like these.

   SEE ALSO
       OpenWorkBench()
```

```
intuition.library/CurrentTime                     intuition.library/CurrentTime

   NAME
       CurrentTime -- Get the current time values.

   SYNOPSIS
       CurrentTime( Seconds, Micros )
                    A0       A1

       VOID CurrentTime( ULONG *, ULONG * );

   FUNCTION
       Puts copies of the current time into the supplied argument pointers.

       This time value is not extremely accurate, nor is it of a very fine
       resolution.  This time will be updated no more than sixty times a
       a second, and will typically be updated far fewer times a second.

   INPUTS
       Seconds = pointer to a LONG variable to receive the current seconds
           value
       Micros = pointer to a LONG variable for the current microseconds value

   RESULT
       Puts the time values into the memory locations specified by the
           arguments
       Return value is not defined.

   BUGS

   SEE ALSO
       timer.device/TR_GETSYSTIME
```

```
intuition.library/DisplayAlert                   intuition.library/DisplayAlert

    NAME
        DisplayAlert -- Create the display of an alert message.

    SYNOPSIS
        Response = DisplayAlert( AlertNumber, String, Height )
        D0                       D0           A0      D1

        BOOL DisplayAlert( ULONG, UBYTE *, WORD );

    FUNCTION
        Creates an alert display with the specified message.

        If the system can recover from this alert, it's a RECOVERY_ALERT and
        this routine waits until the user presses one of the mouse buttons,
        after which the display is restored to its original state and a
        BOOL value is returned by this routine to specify whether or not
        the user pressed the LEFT mouse button.

        If the system cannot recover from this alert, it's a DEADEND_ALERT
        and this routine returns immediately upon creating the alert display.
        The return value is FALSE.

        NOTE: Starting with V33, if Intuition can't get enough
        memory for a RECOVERY_ALERT, the value FALSE will be returned.

        AlertNumber is a LONG value, historically related to the value
        sent to the Alert() routine.  But the only bits that are pertinent to
        this routine are the ALERT_TYPE bit(s).  These bits must be set to
        either RECOVERY_ALERT for alerts from which the system may safely
        recover, or DEADEND_ALERT for those fatal alerts.  These states are
        described in the paragraph above.  There is a third type of
        alert, the DAISY_ALERT, which is used only by the Exec.

        The string argument points to an AlertMessage string.  The AlertMessage
        string is comprised of one or more substrings, each of which is
        composed of the following components:
           - first, a 16-bit x-coordinate and an 8-bit y-coordinate,
             describing where on the alert display you want this string
             to appear.  The y-coordinate describes the offset to the
             baseline of the text.
           - then, the bytes of the string itself, which must be
             null-terminated (end with a byte of zero)
           - lastly, the continuation byte, which specifies whether or
             not there's another substring following this one.  If the
             continuation byte is non-zero, there IS another substring
             to be processed in this alert message.  If the continuation
             byte is zero, this is the last substring in the message.

        The last argument, Height, describes how many video lines tall you
        want the alert display to be.

        New for V36: Alerts are always rendered in Topaz 8 (80 column font),
        regardless of the system default font.  Also, RECOVERY_ALERTs are
        displayed in amber, while DEADEND_ALERTs are still red.  Alerts
        no longer push down the application screens to be displayed.  Rather,
        they appear alone in a black display.

        Also new for V36: Alerts block each other out, and input
        during an alert is deprived of the rest of the system.  Internal
        input buffers still cause alert clicks to be processed by
        applications sometimes.

    INPUTS
        AlertNumber = the number of this alert message.  The only pertinent
```

```
            bits of this number are the ALERT_TYPE bit(s).  The rest of the
           number is ignored by this routine.
        String = pointer to the alert message string, as described above
        Height = minimum display lines required for your message

    RESULT
        A BOOL value of TRUE or FALSE.  If this is a DEADEND_ALERT, FALSE
        is always the return value.  If this is a RECOVERY_ALERT. The return
        value will be TRUE if the user presses the left mouse button in
        response to your message, and FALSE if the user presses the right hand
        button is response to your text, or if the alert could not
        be posted.

    BUGS
        If the system is worse off than you think, the level of your alert
        may become DEADEND_ALERT without you ever knowing about it.  This
        will NOT happen due simply to low memory.  Rather, the alert
        display will be skipped, and FALSE will be returned.

        The left and right button clicks satisfying the alerts are
        unfortunately passed to the normal applications, because of
        some internal system input buffering.

    SEE ALSO
```

```
intuition.library/DisplayBeep                   intuition.library/DisplayBeep

   NAME
       DisplayBeep -- Flash the video display.

   SYNOPSIS
       DisplayBeep( Screen )
                    A0

       VOID DisplayBeep( struct Screen * );

   FUNCTION
       "Beeps" the video display by flashing the background color of the
       specified screen.  If the screen argument is NULL, every screen
       in the display will be beeped.  Flashing everyone's screen is not
       a polite thing to do, so this should be reserved for dire
       circumstances.

       The reason such a routine is supported is because the Amiga has
       no internal bell or speaker.  When the user needs to know of
       an event that is not serious enough to require the use of a requester,
       the DisplayBeep() function may be called.

       New for V36:  Intuition now calls DisplayBeep through the
       external library vector.  This means that if you call SetFunction()
       to replace DisplayBeep with an audible beep, for example, then
       your change will affect even Intuition's calls to DisplayBeep.

   INPUTS
       Screen = pointer to a screen.  If NULL, every screen in the display
          will be flashed

   RESULT
       None

   BUGS

   SEE ALSO
```

```
intuition.library/DisposeObject               intuition.library/DisposeObject

   NAME
       DisposeObject -- Deletes a 'boopsi' object. (V36)

   SYNOPSIS
       DisposeObject( Object )
                      A0

       VOID DisposeObject( APTR );

   FUNCTION
       Deletes a boopsi object and all of it auxiliary data.
       These objects are all created by NewObject().  Objects
       of certain classes "own" other objects, which will also
       be deleted when the object is passed to DisposeObject().
       Read the per-class documentation carefully to be aware
       of these instances.

   INPUTS
       Object = abstract pointer to a boopsi object returned by NewObject()

   NOTES
       This function invokes the OM_DISPOSE method.

   RESULT
       None.

   BUGS

   SEE ALSO
       NewObject(), SetAttrs(), GetAttr(), MakeClass(),
       Document "Basic Object-Oriented Programming System for Intuition"
       and the "boopsi Class Reference" document.
```

```
intuition.library/DoubleClick                        intuition.library/DoubleClick

    NAME
        DoubleClick -- Test two time values for double-click timing.

    SYNOPSIS
        IsDouble = DoubleClick( StartSecs, StartMicros,
        D0                      D0         D1
                CurrentSecs, CurrentMicros )
                D2           D3

        BOOL DoubleClick( ULONG, ULONG, ULONG, ULONG );

    FUNCTION
        Compares the difference in the time values with the double-click
        timeout range that the user has set (using the "Preferences" tool) or
        some other program has configured into the system.  If the
        difference between the specified time values is within the current
        double-click time range, this function returns TRUE, else it
        returns FALSE.

       These time values can be found in input events and IDCMP messages.
       The time values are not perfect; however, they are precise enough for
       nearly all applications.

    INPUTS
        StartSeconds, StartMicros = the timestamp value describing the start of
          the double-click time period you are considering
        CurrentSeconds, CurrentMicros = the timestamp value describing
          the end of the double-click time period you are considering

    RESULT
        If the difference between the supplied timestamp values is within the
        double-click time range in the current set of Preferences, this
        function returns TRUE, else it returns FALSE

    BUGS

    SEE ALSO
        CurrentTime()
```

```
intuition.library/DrawBorder                          intuition.library/DrawBorder

    NAME
        DrawBorder -- Draw the specified Border structure into a RastPort.

    SYNOPSIS
        DrawBorder( RastPort, Border, LeftOffset, TopOffset )
                    A0        A1      D0          D1

        VOID DrawBorder( struct RastPort *, struct Border *, WORD, WORD );

    FUNCTION
        First, sets up the draw mode and pens in the RastPort according to the
        arguments of the Border structure.  Then, draws the vectors of
        the border argument into the RastPort, offset by the left and top
        offsets.

        As with all graphics rendering routines, the border will be clipped to
        to the boundaries of the RastPort's layer, if it exists.  This is
        the case with window RastPorts.

        This routine will draw all borders in the NULL-terminated list linked
        by the NextBorder field of the border argument.

    INPUTS
        RastPort = pointer to the RastPort to receive the border rendering
        Border = pointer to a Border structure
        LeftOffset = the offset to be added to each vector's x coordinate
        TopOffset = the offset to be added to each vector's y coordinate

    RESULT
        None

    BUGS

    SEE ALSO
```

```
intuition.library/DrawImage                        intuition.library/DrawImage

    NAME
        DrawImage -- Draw the specified Image structure into a RastPort.

    SYNOPSIS
        DrawImage( RastPort, Image, LeftOffset, TopOffset )
                   A0        A1     D0          D1

        VOID DrawImage( struct RastPort *, struct Image *, WORD, WORD );

    FUNCTION
        First, sets up the draw mode and pens in the RastPort according to the
        arguments of the Image structure.  Then, moves the image data of
        the image argument into the RastPort, offset by the left and top
        offsets.

        This routine does window layer clipping if you pass your window's
        (layered) RastPort -- if you draw an image outside of your window,
        your imagery will be clipped at the window's edge.  If you pass
        a (non-layered) screen RastPort, you MUST be sure your image is
        wholly contained within the rastport bounds.

        If the NextImage field of the image argument is non-NULL,
        the next image is rendered as well, and so on until some
        NextImage field is found to be NULL.

    INPUTS
        RastPort = pointer to the RastPort to receive image rendering
        Image = pointer to an image structure
        LeftOffset = the offset which will be added to the image's x coordinate
        TopOffset = the offset which will be added to the image's y coordinate

    RESULT
        None

    NOTES
        Intuition always has and will continue to assume there are
        at least as many planes of data pointed to by ImageData as there
        are '1' bits in the PlanePick field.  Please ensure that
        this is so.  (See the intuition.h include file for full details
        on using PlanePick).

    BUGS

    SEE ALSO
        DrawImageState(), EraseImage()
```

```
intuition.library/DrawImageState              intuition.library/DrawImageState

    NAME
        DrawImageState -- Draw an (extended) Intuition Image with
                special visual state. (V36)

    SYNOPSIS
        DrawImageState( RPort, Image, LeftOffset, TopOffset, State, DrawInfo )
                        A0     A1     D0          D1         D2     A2

        VOID DrawImageState( struct RastPort *, struct Image *,
                WORD, WORD, ULONG, struct DrawInfo * );

    FUNCTION
        This function draws an Intuition Image structure in a variety of
        "visual states," which are defined by constants in
        intuition/imageclass.h.  These include:
        IDS_NORMAL              - like DrawImage()
        IDS_SELECTED            - represents the "selected state" of a Gadget
        IDS_DISABLED            - the "ghosted state" of a gadget
        IDS_BUSY                - for future functionality
        IDS_INDETERMINATE       - for future functionality
        IDS_INACTIVENORMAL      - for gadgets in window border
        IDS_INACTIVESELECTED    - for gadgets in window border
        IDS_INACTIVEDISABLED    - for gadgets in window border

        Only IDS_NORMAL will make sense for traditional Image structures,
        this function is more useful when applied to new custom images
        or "object-oriented image classes."

        Each class of custom images is responsible for documenting which
        visual states it supports, and you typically want to use images
        which support the appropriate states with your custom gadgets.

        The DrawInfo parameter provides information invaluable to
        "rendered" images, such as pen color and resolution.  Each
        image class must document whether this parameter is required.

    INPUTS
        RPort    - RastPort for rendering
        Image    - pointer to a (preferably custom) image
        LeftOffset,RightOffset - positional offsets in pixels
        State    - visual state selected from above
        DrawInfo - pointer to packed of pen selections and resolution.

    RESULT
        None.

    EXAMPLE
        Provided separately in the DevCon '90 disk set.

    NOTES


    BUGS


    SEE ALSO
        DrawImage(), GetScreenDrawInfo(), intuition/imageclass.h
```

```
intuition.library/EasyRequestArgs          intuition.library/EasyRequestArgs

    NAME
        EasyRequestArgs -- Easy alternative to AutoRequest(). (V36)
        EasyRequest -- Varargs stub for EasyRequestArgs(). (V36)

    SYNOPSIS
        num = EasyRequestArgs( Window, easyStruct, IDCMP_ptr, ArgList )
        D0                     A0      A1          A2         A3

        LONG EasyRequestArgs( struct Window *, struct EasyStruct *,
                ULONG *, APTR );

        num = EasyRequest( Window, easyStruct, IDCMP_ptr, Arg1, Arg2, ... )

        LONG EasyRequest( struct Window *, struct EasyStruct *,
                ULONG *, APTR, ... );

        ( from intuition.h )
        struct EasyStruct {
            ULONG       es_StructSize;
            ULONG       es_Flags;
            UBYTE       *es_Title;
            UBYTE       *es_TextFormat;
            UBYTE       *es_GadgetFormat;
        };

    FUNCTION
        This function provides a simpler method of using a 'System
        Requester' than provided by AutoRequest().  It performs layout
        and size calculations sensitive to the current font and screen
        resolution.

        It provides for the descriptive 'body' text and the gadget
        text to be constructed from 'printf' style format strings.

        It also provides a general way for the requester to be
        sensitive to particular IDCMP messages.

        The first function listed is the actual Intuition library
        function.  It is passed the arguments for the formatting
        operations as a pointer to the first argument.

        The second function uses a C-style variable number of argument
        (varargs) calling convention.  It should be implemented as
        a call to the first function, and might be supplied by your
        compiler vendor, in amiga.lib, or using the first example below,
        for most C compilers.

        NOTE: The formatting is done by exec.library/RawDoFmt(), so
        be aware that to display a 32-bit integer argument, for
        example, you must say "%ld", not "%d", since RawDoFmt() is
        "word-oriented."

        NOTE: This function switches the processor stack to ensure
        sufficient stack space for the function to complete.

    EXAMPLES
        /* varargs interface works for most C compilers */
        EasyRequest( w, es, ip, arg1 )
        struct Window   *w;
        struct EasyStruct *es;
        ULONG           *ip;
        int             arg1;
        {
            return ( EasyRequestArgs( w, es, ip, &arg1 ) );
```

```
        }

        /*********************************************/

        /* typical use */
        struct EasyStruct volumeES = {
            sizeof (struct EasyStruct),
            0,
            "Volume Request",
            "Please insert volume %s in any drive.",
            "Retry|Cancel",
        };
        #define CANCEL  (0)

        Volume  *
        getVolume( volname )
        UBYTE   *volname;
        {
            Volume      *vptr;
            Volume      *findVolume();
            UWORD       reply;
            ULONG       iflags;

            iflags = IDCMP_DISKINSERTED;

        while ( ((vptr = findVolume( volname )) == NULL) &&
         (EasyRequest( w, &volumeES, &iflags, volname ) != CANCEL) )
                /* loop */ ;

            /* note that in some circumstances, you will have to
               re-initialize the value of 'iflags'.  Here, it
               is either unchanged, or returned as the single
               IDCMPFlag value IDCMP_DISKINSERTED.  If you combine
               multiple  IDCMPFlag values in 'iflags,' only
               one will be returned, so you must reinitialize
               'iflags' to be the combination.
              */
            return ( vptr );
        }

    INPUTS
        Window = Reference window pointer, determines the screen and
            title of the requester window.  This can be NULL, which
            means the requester is to appear on the Workbench screen,
            or default public screen, if defined.
        IDCMP_ptr = Pointer to IDCMP flags that you want to terminate
                the requester.  This pointer may be NULL.
        easyStruct = Pointer to EasyStruct structure with fields
                interpreted as follows:

        es_StructSize = sizeof (struct EasyStruct), for future extension.
        es_Flags = 0 for now, in the future may specify other options.
        es_Title = Title of system requester window.  If this is NULL,
            the title will be taken to be the same as the title of 'Window',
            if provided, or else "System Request."
        es_TextFormat = Format string, a la RawDoFmt(), for message in
            requester body.  Lines are separated by '\n'.  Formatting
            '%' functions are supported exactly as in RawDoFmt().
        es_GadgetFormat = Format string for gadgets.  Text for separate
            gadgets is separated by '|'.  Format functions are supported.
            You MUST specify at least one gadget.

        Args = Arguments for format commands.   Arguments for
            GadFmt follow arguments for TextFmt.

    RESULT
    0, 1, ..., N = Successive GadgetID values, for the gadgets
        you specify for the requester.  NOTE: The numbering
        from left to right is actually: 1, 2, ..., N, 0.
        This is for compatibility with AutoRequests which has
        FALSE for the rightmost gadget.

    -1 = Means that one of the caller-supplied IDCMPFlags occurred.
        The IDCMPFlag value is in the longword pointed to by IDCMP_ptr.

BUGS
    Does not fall back to a recoverable alert if the requester
     cannot be created.

    Does not handle case when gadgets don't fit or window title
    is too long, although it does trim trailing spaces from the
    title for calculating dimensions.

PLANS
    Possible enhancements include: centering of text, size-sensitive
    layout,  window-relative requester, vertical gadget layout,
    window placement, more keyboard shortcuts.

    We also reserve the use of the newline character '\n' in
    gadget format strings for future use as a line separator.

SEE ALSO
    exec.library/RawDoFmt(), BuildEasyRequestArgs(), SysReqHandler(),
    AutoRequest(), BuildSysRequest()
```

```
intuition.library/EndRefresh                      intuition.library/EndRefresh

    NAME
        EndRefresh -- End the optimized refresh state of the window.

    SYNOPSIS
        EndRefresh( Window, Complete )
                    A0      D0

        VOID EndRefresh( struct Window *, BOOL );

    FUNCTION
        This function gets you out of the special refresh state of your
        window.  It is called following a call to BeginRefresh(), which
        routine puts you into the special refresh state.  While your window
        is in the refresh state, the only rendering that will be wrought in
        your window will be to those areas which were recently revealed and
        need to be refreshed.

        After you've done all the refreshing you want to do for this window,
        you should call this routine to restore the window to its
        non-refreshing state.  Then all rendering will go to the entire
        window, as usual.

        The 'Complete' argument is a boolean TRUE or FALSE value used to
        describe whether or not the refreshing you've done was all the
        refreshing that needs to be done at this time.  Most often, this
        argument will be TRUE.  But if, for instance, you have multiple
        tasks or multiple procedure calls which must run to completely
        refresh the window, then each can call its own Begin/EndRefresh()
        pair with a Complete argument of FALSE, and only the last calls
        with a Complete argument of TRUE.

        WARNING:  Passing this function the value of FALSE has its
        pitfalls.  Please see the several caveats in the autodoc for
        BeginRefresh().

        For your information, this routine calls the Layers library function
        EndUpdate(), unlocks your layers (calls UnlockLayerRom()), clears
        the LAYERREFRESH bit in your Layer Flags, and clears the
        WFLG_WINDOWREFRESH bit in your window Flags.

    INPUTS
        Window = pointer to the window currently in optimized-refresh mode
        Complete = Boolean TRUE or FALSE describing whether or not this
           window is completely refreshed

    RESULT
        None

    BUGS

    SEE ALSO
        BeginRefresh(), layers.library/EndUpdate(),
        graphics.library/UnlockLayerRom()
```

intuition.library Page 39

```
intuition.library/EndRequest                    intuition.library/EndRequest

   NAME
       EndRequest -- Remove a currently active requester.

   SYNOPSIS
       EndRequest( Requester, Window )
                   A0         A1

       VOID EndRequest( struct Requester *, struct Window * );

   FUNCTION
       Ends the request by erasing the requester and decoupling it from
       the window.

       Note that this doesn't necessarily clear all requesters from the window,
       only the specified one.  If the window labors under other requesters,
       they will remain in the window.

   INPUTS
       Requester = pointer to the requester to be removed
       Window = pointer to the Window structure with which this requester
          is associated

   RESULT
       None

   BUGS

   SEE ALSO
       Request()
```

intuition.library Page 40

```
intuition.library/EraseImage                    intuition.library/EraseImage

   NAME
       EraseImage -- Erases an Image. (V36)

   SYNOPSIS
       EraseImage( RPort, Image, LeftOffset, TopOffset )
                   A0     A1     D0          D1

       VOID EraseImage( struct RastPort *, struct Image *, WORD, WORD );

   FUNCTION
       Erases an Image.  For a normal Image structure, this will
       call the graphics function EraseRect() (clear using layer
       backfill, if any) for the Image box (LeftEdge/TopEdge/Width/Height).

       For custom image, the exact behavior is determined by the
       custom image class.

   INPUTS
       RPort   - RastPort to erase a part of
       Image   - custom or standard image
       LeftOffset,RightOffset - pixel offsets of Image position

   RESULT
       None.

   EXAMPLE


   NOTES


   BUGS


   SEE ALSO
       graphics.library/EraseRect().
```

```
intuition.library/FreeClass                         intuition.library/FreeClass

    NAME
        FreeClass -- Frees a boopsi class created by MakeClass(). (V36)

    SYNOPSIS
        success = FreeClass( ClassPtr )
        D0                   A0

        BOOL FreeClass( struct IClass * );

    FUNCTION
        For class implementors only.

        Tries to free a boopsi class created by MakeClass().  This
        won't always succeed: classes with outstanding objects or
        with subclasses cannot be freed.  You cannot allow the code
        which implements the class to be unloaded in this case.

        For public classes, this function will *always* remove
        the class (see RemoveClass() ) making it unavailable, whether
        it succeeds or not.

        If you have a dynamically allocated data for your class (hanging
        off of cl_UserData), try to free the class before you free the
        user data, so you don't get stuck with a half-freed class.

    INPUTS
        ClassPtr - pointer to a class created by MakeClass().

    RESULT
        Returns FALSE if the class could not be freed.  Reasons include,
        but will not be limited to, having non-zero cl_ObjectCount or
        cl_SubclassCount.

        Returns TRUE if the class could be freed.

        Calls RemoveClass() for the class in either case.

    EXAMPLE
        Freeing a private class with dynamically allocated user data:

        freeMyClass( cl )
        struct IClass   *cl;
        {
            struct MyPerClassData       *mpcd;

            mpcd = (struct MyPerClassData *) cl->cl_UserData;
            if ( FreeClass( cl ) )
            {
                FreeMem( mpcd, sizeof mpcd );
                return ( TRUE );
            }
            else
            {
                return ( FALSE );
            }
        }

    BUGS

    SEE ALSO
        MakeClass(),
        Document "Basic Object-Oriented Programming System for Intuition"
        and the "boopsi Class Reference" document.
```

```
intuition.library/FreeRemember                   intuition.library/FreeRemember

    NAME
        FreeRemember -- Free memory allocated by calls to AllocRemember().

    SYNOPSIS
        FreeRemember( RememberKey, ReallyForget )
                      A0           D0

        VOID FreeRemember( struct Remember **, BOOL );

    FUNCTION
        This function frees up memory allocated by the AllocRemember() function.
        It will either free up just the Remember structures, which supply the
        link nodes that tie your allocations together, or it will deallocate
        both the link nodes AND your memory buffers too.

        If you want to deallocate just the Remember structure link nodes,
        you should set the ReallyForget argument to FALSE.  However, if you
        want FreeRemember to really deallocate all the memory, including
        both the Remember structure link nodes and the buffers you requested
        via earlier calls to AllocRemember(), then you should set the
        ReallyForget argument to TRUE.

        NOTE WELL: Once you call this function passing it FALSE, the
        linkages between all the memory chunks are lost, and you
        cannot subsequently use FreeRemember() to free them.

    INPUTS
        RememberKey = the address of a pointer to struct Remember.  This
           pointer should either be NULL or set to some value (possibly
           NULL) by a call to AllocRemember().
        ReallyForget = a BOOL FALSE or TRUE describing, respectively,
           whether you want to free up only the Remember nodes or
           if you want this procedure to really forget about all of
           the memory, including both the nodes and the memory buffers
           referenced by the nodes.

    EXAMPLE
        struct Remember *RememberKey;
        RememberKey = NULL;
        AllocRemember(&RememberKey, BUFSIZE, MEMF_CHIP);
        FreeRemember(&RememberKey, TRUE);

    RESULT
        None

    BUGS

    SEE ALSO
        AllocRemember(), exec.library/FreeMem()
```

```
intuition.library/FreeScreenDrawInfo     intuition.library/FreeScreenDrawInfo

   NAME
       FreeScreenDrawInfo -- Finish using a DrawInfo structure. (V36)

   SYNOPSIS
       FreeScreenDrawInfo( Screen, DrInfo )
                           A0      A1

       VOID FreeScreenDrawInfo( struct Screen *, struct DrawInfo * );

   FUNCTION
       Declares that you are finished with the DrawInfo structure
       returned by GetScreenDrawInfo().

   INPUTS
       Screen          - pointer to screen passed to GetScreenDrawInfo()
       DrInfo        - pointer to DrawInfo returned by GetScreenDrawInfo()

   RESULT
       None

   NOTES
       This function, and GetScreenDrawInfo(), don't really do much, but
       they provide an upward compatibility path.  That means that
       if you misuse them today, they probably won't cause a problem,
       although they may someday later.  So, please be very careful
       only to use the DrawInfo structure between calls to
       GetScreenDrawInfo() and FreeScreenDrawInfo(), and be sure
       that you don't forget FreeScreenDrawInfo().

   BUGS

   SEE ALSO
       GetScreenDrawInfo()
```

```
intuition.library/FreeSysRequest             intuition.library/FreeSysRequest

   NAME
       FreeSysRequest -- Free resources gotten by a call to BuildSysRequest().

   SYNOPSIS
       FreeSysRequest( Window )
                       A0

       VOID FreeSysRequest( struct Window * );

   FUNCTION
       This routine frees up all memory allocated by a successful call to
       the BuildSysRequest() procedure.  If BuildSysRequest() returned a
       pointer to a window, then you are able to wait on the message port
       of that window to detect an event which satisfies the requester.
       When you want to remove the requester, you call this procedure.  It
       ends the requester and deallocates any memory used in the creation
       of the requester.  It also closes the special window that was opened
       for your system requester.

       For V36: It's OK if you pass a NULL or a TRUE (1) value to
       this function.  Also, this function properly disposes of
       requesters gotten using BuildEasyRequest().

   INPUTS
       Window = value of the window pointer returned by a successful call to
          the BuildSysRequest() procedure

   RESULT
       None

   BUGS

   SEE ALSO
       BuildSysRequest(), AutoRequest(), CloseWindow()
```

```
intuition.library/GadgetMouse                    intuition.library/GadgetMouse

   NAME
       GadgetMouse -- Calculate gadget-relative mouse position. (V36)

   SYNOPSIS
       GadgetMouse( Gadget, GInfo, MousePoint )
                    A0      A1     A2

       VOID GadgetMouse( struct GadgetInfo *, WORD * );

   FUNCTION
       Determines the current location of the mouse pointer relative
       to the upper-left corner of a custom gadget.  Typically used
       only in the GM_HANDLEINPUT and GM_GOACTIVE custom gadget hook
       routines.

       NEWS FLASH!!: These two hook routines are now passed the mouse
       coordinates, so this function has no known usefulness.

       We recommend that you don't call it.

       Note that this function calculates the mouse position taking
       "gadget relativity" (GFLG_RELRIGHT, GFLG_RELBOTTOM) into
       consideration.  If your custom gadget intends to ignore these
       properties, then you should either enjoin or inhibit your users
       from setting those bits, since Intuition won't ask if you respect
       them.

   INPUTS
       GInfo = A pointer to a GadgetInfo structure as passed to the
               custom gadget hook routine.
       MousePoint = address of two WORDS, or a pointer to a structure of
               type Point.

   RESULT
       Returns nothing.  Fills in the two words pointed to by
       MousePoint with the gadget-relative mouse position.

   BUGS
       Useless, since equivalent information is now passed to every
       function that might have a use for this.

   SEE ALSO
```

```
intuition.library/GetAttr                                intuition.library/GetAttr

   NAME
       GetAttr -- Inquire the value of some attribute of an object. (V36)

   SYNOPSIS
       attr = GetAttr( AttrID, Object, StoragePtr )
       D0              D0      A0      A1

       ULONG GetAttr( ULONG, APTR, ULONG * );

   FUNCTION
       Inquires from the specified object the value of the specified attribute.

       You always pass the address of a long variable, which will
       receive the same value that would be passed to SetAttrs() in
       the ti_Data portion of a TagItem element.  See the documentation
       for the class for exceptions to this general rule.

       Not all attributes will respond to this function.  Those that
       will are documented on a class-by-class basis.

   INPUTS
       AttrID = the attribute tag ID understood by the object's class
       Object = abstract pointer to the boopsi object you are interested in
       StoragePtr = pointer to appropriate storage for the answer

   RESULT
       Returns FALSE (0) if the inquiries of attribute are not provided
       by the object's class.

   NOTES
       This function invokes the OM_GET method of the object.

   BUGS

   SEE ALSO
       NewObject(), DisposeObject(), SetAttrs(), MakeClass(),
       Document "Basic Object-Oriented Programming System for Intuition"
       and the "boopsi Class Reference" document.
```

intuition.library Page 47

```
intuition.library/GetDefaultPubScreen   intuition.library/GetDefaultPubScreen

    NAME
        GetDefaultPubScreen -- Get name of default public screen. (V36)

    SYNOPSIS
        GetDefaultPubScreen( Namebuff )
                             A0

        VOID GetDefaultPubScreen( UBYTE * );

    FUNCTION
        Provides the name of the current default public screen.
        Only anticipated use is for Public Screen Manager utilities,
        since it is easy to open a visitor window on the default
        public screen without specifying the name.

    INPUTS
        Namebuff = a buffer of MAXPUBSCREENNAME.  This can be NULL.

    RESULT
        None.  Will provide the string "Workbench" in Namebuff if there
        is no current default public screen.

    NOTE
        This function actually "returns" in register D0 a pointer
        to the public screen.  Unfortunately, the lifespan of
        this pointer is not ensured; the screen could be closed
        at any time.  The *ONLY* legitimate use we can see for
        this return value is to compare for identity with the pointer
        to a public screen you either have a window open in, or
        a lock on using LockPubScreen(), to determine if that
        screen is in fact the default screen.

    BUGS
        The function prototype does not reflect the return value.

    SEE ALSO
        SetDefaultPubScreen(), OpenWindow()
```

intuition.library Page 48

```
intuition.library/GetDefPrefs                     intuition.library/GetDefPrefs

    NAME
        GetDefPrefs -- Get a copy of the the Intuition default Preferences.

    SYNOPSIS
        Prefs = GetDefPrefs( PrefBuffer, Size )
        D0                   A0          D0

        struct Preferences *GetDefPrefs( struct Preferences *, WORD );

    FUNCTION
        Gets a copy of the Intuition default preferences data.  Writes the
        data into the buffer you specify.  The number of bytes you want
        copied is specified by the size argument.

        The default preferences are those that Intuition uses when it
        is first opened.  If no preferences file is found, these are
        the preferences that are used.  These would also be the startup
        preferences in an AmigaDOS-less environment.

        It is legal to take a partial copy of the Preferences structure.
        The more pertinent preferences variables have been grouped near
        the top of the structure to facilitate the memory conservation
        that can be had by taking a copy of only some of the Preferences
        structure.

    INPUTS
        PrefBuffer = pointer to the memory buffer to receive your copy of the
                Intuition Preferences structure
        Size = the number of bytes in your PrefBuffer, the number of bytes
          you want copied from the system's internal Preference settings

    RESULT
        Returns your parameter PrefBuffer.

    BUGS

    SEE ALSO
        GetPrefs()
```

```
intuition.library/GetPrefs                          intuition.library/GetPrefs

    NAME
        GetPrefs -- Get the current Intuition Preferences structure.

    SYNOPSIS
        Prefs = GetPrefs( PrefBuffer, Size )
        D0                A0          D0

        struct Preferences *GetPrefs( struct Preferences *, WORD );

    FUNCTION
        Gets a copy of the current Intuition Preferences structure.
        Writes the data into the buffer you specify.  The number of bytes you
        want copied is specified by the size argument.

        It is legal to take a partial copy of the Preferences structure.
        The more pertinent preferences variables have been grouped near
        the top of the structure to facilitate the memory conservation
        that can be had by taking a copy of only some of the Preferences
        structure.

        New for V36:  A new and more extensible method for supplying
        Preferences has been introduced in V36, and relies on file
        system notification.  The Intuition preferences items rely
        also on the IPrefs program.  Certain elements of the
        Preferences structure have been superceded by this new method.
        As much as possible, the Preferences structure returned by
        GetPrefs() reflect the current state of Preferences.  However,
        it is impossible to represent some of the V36-style preferences
        items using the existing Preferences structure.

    INPUTS
        PrefBuffer = pointer to the memory buffer to receive your copy of the
           Intuition Preferences
        Size = the number of bytes in your PrefBuffer, the number of bytes
           you want copied from the system's internal Preference settings

    RESULT
        Returns your parameter PrefBuffer.

    BUGS

    SEE ALSO
        GetDefPrefs(), SetPrefs()
```

```
intuition.library/GetScreenData                intuition.library/GetScreenData

    NAME
        GetScreenData -- Get copy of a screen data structure.

    SYNOPSIS
        Success = GetScreenData( Buffer, Size, Type, Screen )
        D0                       A0      D0    D1    A1

        BOOL GetScreenData( APTR, UWORD, UWORD, struct Screen * );

    FUNCTION
        This function copies into the caller's buffer data from a Screen
        structure.  Typically, this call will be used to find the size, title
        bar height, and other values for a standard screen, such as the
        Workbench screen.

        To get the data for the Workbench screen, one would call:
            GetScreenData(buff, sizeof(struct Screen), WBENCHSCREEN, NULL)

        NOTE: if the requested standard screen is not open, this function
        will have the effect of opening it.

        This function has been useful for two basic types of things:
        1) Determining information about the Workbench screen, in
           preparation for opening a window on it.
        2) Attempts at discerning the user's preferences in a working
           screen, for "cloning" the Workbench modes and dimensions
           when opening a similar custom screen.

        Providing compatibility with both of these goals has proven
        difficult, as we introduce new display modes and screen scrolling
        in V36.  Read carefully the somewhat involved exceptions we
        elected to implement ...

        Changes as of V36:

        For V36 and later, the function LockPubScreen() is an improvement
        over this function, in that it doesn't copy the screen data
        but returns a pointer and a guarantee that the screen will not
        be closed.

        If the global public screen SHANGHAI mode is in effect (see
        SetPubScreenModes() ), this function will actually report on
        the default public screen, where "Workbench" windows will
        actually open.

        For V36 and later, this function does some "compatibility tricks"
        when you inquire about the WBENCHSCREEN.  To keep programs from
        "stumbling" into modes they don't understand, and because an NTSC
        machine may be running a PAL Workbench or PRODUCTIVITY, for example,
        the following "false" information is returned.

        The Screen.ViewPort.Modes field will either be HIRES or HIRES+LACE
        (with the SPRITES flag also set, as usual).  HIRES+LACE is
        used if the display mode selected for the Workbench screen
        is an interlaced screen of any type.

        The dimensions returned will be the *smaller* of the OSCAN_TEXT
        dimensions for the returned mode, and the actual dimensions
        of the Workbench screen.

        EXCEPTION: For specific compatibility considerations, if the
        Workbench is in one of the A2024 modes, the mode returned
        in Screen.ViewPort.Modes will be HIRES+LACE (with perhaps
        some "special" bits also set for future improvement), but
    with dimensions equal to the actual A2024-mode Workbench screen.
    This will favor programs which open windows on the A2024
    Workbench, but will cause some problems for programs which
    try to "clone" the Workbench screen using this function.

    If you want the real information about the modern Workbench
    screen, call LockPubScreen( "Workbench" ) and acquire its
    display mode ID by inquiring of the actual ViewPort (using
    graphics.library/GetVPModeID() ).

    You may then use the information you get to clone as many of
    the properties of the Workbench screen that you wish.

    In the long run, it's probably better to provide your user
    with a screen mode selection option, and skip all this.

INPUTS
    Buffer = pointer to a buffer into which data can be copied
    Size   = the size of the buffer provided, in bytes
    Type   = the screen type, as specified in OpenWindow() (WBENCHSCREEN,
        CUSTOMSCREEN, ...)
    Screen = ignored, unless type is CUSTOMSCREEN, which results only in
        copying 'size' bytes from 'screen' to 'buffer'

RESULT
    TRUE if successful
    FALSE if standard screen of Type 'type' could not be opened.

BUGS
    You cannot support the new V36 display modes using this function.

SEE ALSO
    OpenWindow(), LockPubScreen(), graphics.library/GetVPModeID(),
    SetPubScreenModes(), OpenScreen()
```

```
intuition.library/GetScreenDrawInfo       intuition.library/GetScreenDrawInfo

   NAME
       GetScreenDrawInfo -- Get pointer to rendering information. (V36)

   SYNOPSIS
       DrInfo = GetScreenDrawInfo( Screen )
       D0                          A0

       struct DrawInfo *GetScreenDrawInfo( struct Screen * );

   FUNCTION
       Returns a pointer to a DrawInfo structure derived from the
       screen passed.  This data structure is READ ONLY.  The field
       dri_Version identifies which version of struct DrawInfo you
       are given a pointer to.

   INPUTS
       Screen        - pointer to a valid, open screen.

   RESULT
       DrInfo - pointer to a system-allocated DrawInfo structure,
       as defined in intuition/screens.h.

   NOTES
       Some information in the DrawInfo structure may in the future
       be calculated the first time this function is called for a
       particular screen.

       You must call FreeScreenDrawInfo() when you are done using the
       returned pointer.

       This function does not prevent a screen from closing.  Apply it
       only to the screens you opened yourself, or apply a protocol
       such as LockPubScreen().

       WARNING: Until further notice, the pointer returned does not
       remain valid after the screen is closed.

       This function and FreeScreenDrawInfo() don't really do much now,
       but they provide an upward compatibility path.  That means that
       if you misuse them today, they probably won't cause a problem,
       although they may someday later.  So, please be very careful
       only to use the DrawInfo structure between calls to
       GetScreenDrawInfo() and FreeScreenDrawInfo(), and be sure
       that you don't forget FreeScreenDrawInfo().

   BUGS
       Does not reflect to changes in screen modes, depth, or pens.

   SEE ALSO
       FreeScreenDrawInfo(), LockPubScreen(), intuition/screens.h
```

```
intuition.library/InitRequester                    intuition.library/InitRequester

    NAME
        InitRequester -- Initialize a Requester structure.

    SYNOPSIS
        InitRequester( Requester )
                       A0

        VOID InitRequester( struct Requester * );

    FUNCTION
        Initializes a requester for general use.  After calling InitRequester,
        you need fill in only those Requester values that fit your needs.
        The other values are set to NULL--or zero--states.

        Note that the example in the early versions of the Intuition
        Reference Manual is flawed because the Requester structure is
        initialized BEFORE InitRequester is called.  Be sure to
        perform your initialization AFTER calling InitRequester.

    INPUTS
        Requester = a pointer to a Requester structure

    RESULT
        None

    BUGS
        Since the publication of the first Intuition Manual to this
        day, most people haven't used this function, and for
        compatibility reasons, we'll never be able to assume that
        they do.  Thus, this function is useless.

    SEE ALSO
```

```
intuition.library/IntuiTextLength                intuition.library/IntuiTextLength

    NAME
        IntuiTextLength -- Return the length (pixel-width) of an IntuiText.

    SYNOPSIS
        Length = IntuiTextLength( IText )
        D0                        A0

        LONG IntuiTextLength( struct IntuiText * );

    FUNCTION
        This routine accepts a pointer to an instance of an IntuiText structure,
        and returns the length (the pixel-width) of the string which that
        instance of the structure represents.

        NOTE: if the Font pointer of your IntuiText structure is set to NULL,
        you'll get the pixel-width of your text in terms of the current system
        default font.  You may wish to be sure that the field IText->ITextFont
        for 'default font' text is equal to the Font field of the screen it is
        being measured for.

    INPUTS
        IText = pointer to an instance of an IntuiText structure

    RESULT
        Returns the pixel-width of the text specified by the IntuiText data

    BUGS
        Would do better to take a RastPort as argument, so that a NULL in
        the Font pointer would lead automatically to the font for the
        intended target RastPort, rather than the system default font.

    SEE ALSO
        OpenScreen()
```

intuition.library

Page 55

```
intuition.library/ItemAddress                         intuition.library/ItemAddress

    NAME
        ItemAddress -- Returns the address of the specified MenuItem.

    SYNOPSIS
        Item = ItemAddress( MenuStrip, MenuNumber )
        D0                  A0         D0

        struct MenuItem *ItemAddress( struct Menu *, UWORD );

    FUNCTION
        This routine feels through the specified menu strip and returns the
        address of the item specified by the menu number.  Typically,
        you will use this routine to get the address of a menu item from
        a menu number sent to you by Intuition after user has chosen from
        a window's menus.

       This routine requires that the arguments are well-defined.
        MenuNumber may be equal to MENUNULL, in which case this routine returns
        NULL.  If MenuNumber doesn't equal MENUNULL, it's presumed to be a
        valid item number selector for your menu strip, which includes:
            - a valid menu number
            - a valid item number
            - if the item specified by the above two components has a
              sub-item, the menu number may have a sub-item component, too.

        Note that there must be BOTH a menu number and an item number.
        Because a sub-item specifier is optional, the address returned by
        this routine may point to either an item or a sub-item.

    INPUTS
        MenuStrip = a pointer to the first menu in your menu strip
        MenuNumber = the value which contains the packed data that selects
           the menu and item (and sub-item).  See the Intuition Reference
            Manual for information on menu numbers.

    RESULT
        If MenuNumber == MENUNULL, this routine returns NULL,
        else this routine returns the address of the menu item specified
        by MenuNumber.

    BUGS

    SEE ALSO
        The "Menus" chapter of the Intuition Reference Manual,
        or the Amiga Rom Kernel Manual
```

intuition.library

Page 56

```
intuition.library/LockIBase                             intuition.library/LockIBase

    NAME
        LockIBase -- Invoke semaphore arbitration of IntuitionBase.

    SYNOPSIS
        Lock = LockIBase( LockNumber )
        D0                D0

        ULONG LockIBase( ULONG );

    FUNCTION
        Grabs Intuition internal semaphore so that caller may examine
        IntuitionBase safely.  This function is not a magic "fix all my
        race conditions" panacea.

        The idea here is that you can get the locks Intuition needs before
        such IntuitionBase fields as ActiveWindow and FirstScreen are
        changed, or linked lists of windows and screens are changed.

        Do Not Get Tricky with this entry point, and do not hold these locks
        for long, as all Intuition input processing will wait for you to
        surrender the lock by a call to UnlockIBase().

        NOTE WELL: A call to this function MUST be paired with a subsequent
        call to UnlockIBase(), and soon, please.

        NOTE WELL: Do not call any Intuition functions (nor any graphics,
        layers, dos, or other high-level system function) while
        holding this lock.

    INPUTS
        A long unsigned integer, LockNumber, specifies which of Intuition's
        internal locks you want to get.  This parameter should be zero for all
        forseeable uses of this function, which will let you examine active
        fields and linked lists of screens and windows with safety.

    RESULT
        Returns another ULONG which should be passed to UnlockIBase() to
        surrender the lock gotten by this call.

    BUGS
        This function must not be called while holding any other system locks
        such as layer or LayerInfo locks.

    SEE ALSO
        UnlockIBase(), layers.library/LockLayerInfo(),
        exec.library/ObtainSemaphore()
```

```
intuition.library/LockPubScreen                 intuition.library/LockPubScreen

    NAME
        LockPubScreen -- Prevent a public screen from closing. (V36)

    SYNOPSIS
        screen = LockPubScreen( Name )
        D0                      A0

        struct Screen *LockPubScreen( UBYTE * );

    FUNCTION
        Prevents a public screen (or the Workbench) from closing
        while you examine it in preparation of opening a visitor window.

        The sequence you use to open a visitor window that needs to
        examine fields in the screen it is about to open on is:
                LockPubScreen()
                ... examine fields ...
                OpenWindow() on public screen
                UnlockPubScreen()
                ... use your window ...
                CloseWindow()

    NOTE
        You needn't hold the "pubscreen lock" for the duration that
        your window is opened.  LockPubScreen() basically has the
        same effect as an open visitor window: it prevents the
        screen from being closed.

        If you pass the string "Workbench" or you pass NULL and there
        is no default public screen, the Workbench screen will
        be automatically opened if it is not already present.

    INPUTS
        Name = name string for public screen or NULL for default public
        screen.  The string "Workbench" indicates the Workbench
        screen.

    RESULT
        Returns pointer to a screen, if successful, else NULL.
        The call can fail for reasons including that the named
        public screen doesn't exist or is in private state.

    BUGS

    SEE ALSO
        OpenWindow(), UnlockPubScreen(), GetScreenData()
```

```
intuition.library/LockPubScreenList         intuition.library/LockPubScreenList

    NAME
        LockPubScreenList -- Prevent changes to the system list. (V36)

    SYNOPSIS
        List = LockPubScreenList()
        D0

        struct List *LockPubScreenList( VOID );

    FUNCTION
        Arbitrates access to public screen list while you quickly
        make a copy of it for display to the user.

        Note that this is intended only for the Public Screen Manager
        program.

    NOTES
        The nodes on the list are PubScreenNode structures.
        Act quickly while holding this lock.  The restrictions
        on LockIBase() apply here as well.

    INPUTS
        None.

    RESULT
        A pointer to the public screen list.

    BUGS

    SEE ALSO
        OpenScreen(), Intuition V36 update documentation
```

```
intuition.library/MakeClass                          intuition.library/MakeClass

    NAME
        MakeClass -- Create and initialize a boopsi class. (V36)

    SYNOPSIS
        iclass = MakeClass( ClassID, SuperClassID, SuperClassPtr,
        D0                  A0       A1            A2
                InstanceSize, Flags )
                D0            D1

        struct IClass *MakeClass( UBYTE *, UBYTE *, struct IClass *,
                UWORD, ULONG );

    FUNCTION
        For class implementors only.

        This function creates a new public or private boopsi class.
        The superclass should be defined to be another boopsi class:
        all classes are descendants of the class "rootclass".

        Superclasses can be public or private.  You provide a name/ID
        for your class if it is to be a public class (but you must
        have registered your class name and your attribute ID's with
        Commodore before you do this!).  For a public class, you would
        also call AddClass() to make it available after you have
        finished your initialization.

        Returns pointer to an IClass data structure for your
        class.  You then initialize the Hook cl_Dispatcher for
        your class methods code.  You can also set up special data
        shared by all objects in your class, and point cl_UserData at it.
        The last step for public classes is to call AddClass().

        You dispose of a class created by this function by calling
        FreeClass().

    INPUTS
        ClassID = NULL for private classes, the name/ID string for public
                classes
        SuperClassID = name/ID of your new class's superclass.  NULL if
                superclass is a private class
        SuperClassPtr = pointer to private superclass.  Only used if
                SuperClassID is NULL.  You are required never to provide
                a NULL superclass.
        InstanceSize = the size of the instance data that your class's
                objects will require, beyond that data defined for
                your superclass's objects.
        Flags = for future enhancement, including possible additional
                parameters.  Provide zero for now.

    RESULT
        Pointer to the resulting class, or NULL if not possible:
        - no memory for class data structure
        - public superclass not found
        - public class of same name/ID as this one already exists

    NOTES

    EXAMPLE
        Creating a private subclass of a public class:

    /* per-object instance data defined by my class     */
    struct MyInstanceData {
        ULONG   mid_SomeData;
    };
```

```
    /* some useful table I'll share use for all objects */
    UWORD myTable[] = {
        5, 4, 3, 2, 1, 0
    };

    struct IClass       *
    initMyClass()
    {
        ULONG __saveds  myDispatcher();
        ULONG __hookEntry();      /* asm-to-C interface glue      */
        struct IClass   *cl;
        struct IClass   *MakeClass();

        if ( cl =  MakeClass( NULL,
                    SUPERCLASSID, NULL,          /* superclass is public      */
                    sizeof (struct MyInstanceData),
                    0 ))
        {
            /* initialize the cl_Dispatcher Hook        */
            cl->cl_Dispatcher.h_Entry = hookEntry;
            cl->cl_Dispatcher.h_SubEntry = myDispatcher;
            cl->cl_Dispatcher.h_Data = (VOID *) 0xFACE; /* unused */

            cl-cl_UserData = (ULONG) myTable;
        }
        return ( cl );
    }

    BUGS
        The typedef 'Class' isn't consistently used.  Class pointers
        used blindly should be APTR, or struct IClass for class implementors.

    SEE ALSO
        FreeClass(), AddClass(), RemoveClass(), NewObject(),
        Document "Basic Object-Oriented Programming System for Intuition"
        and the "boopsi Class Reference" document.
```

```
intuition.library/MakeScreen                  intuition.library/MakeScreen

   NAME
       MakeScreen -- Do an Intuition-integrated MakeVPort() of a screen.

   SYNOPSIS
       MakeScreen( Screen )
                   A0

       VOID MakeScreen( struct Screen * );

   FUNCTION
       This procedure allows you to do a MakeVPort() for the viewport of your
       custom screen in an Intuition-integrated way.  This way you can
       do your own screen manipulations without worrying about interference
       with Intuition's usage of the same viewport.

       The operation of this function is as follows:
           - Block until the Intuition View structure is not in being changed.
           - Set the view modes correctly to reflect if there is a (visible)
             interlaced screen.
           - call MakeVPort(), passing the Intuition View and your screen's
             ViewPort.
           - Unlocks the Intuition View.

       After calling this routine, you should call RethinkDisplay() to
       incorporate the new viewport of your custom screen into the
       Intuition display.

       NOTE: Intuition may determine that because of a change in global
       interlace needs that all viewports need to be remade, so
       it may effectively call RemakeDisplay().

   INPUTS
       Screen = address of the custom screen structure

   RESULT
       None

   BUGS

   SEE ALSO
       RethinkDisplay(), RemakeDisplay(), graphics.library/MakeVPort()
```

```
intuition.library/ModifyIDCMP                  intuition.library/ModifyIDCMP

   NAME
       ModifyIDCMP -- Modify the state of a window's IDCMPFlags.

   SYNOPSIS
       [Success =] ModifyIDCMP( Window, IDCMPFlags )
       [D0]                     A0      D0

       [BOOL] ModifyIDCMP( struct Window *, ULONG );
       /* returns BOOL in V37 and greater */

   FUNCTION
       This routine modifies the state of your window's IDCMP (Intuition
       Direct Communication Message Port).  The state is modified to reflect
       your desires as described by the flag bits in the value IDCMPFlags.

       The four actions that might be taken are:

     - if there is currently no IDCMP in the given window, and IDCMPFlags
       is zero, nothing happens
     - if there is currently no IDCMP in the given window, and any of the
       IDCMPFlags is selected (set), then the IDCMP of the window is
       created, including allocating and initializing the message ports
       and allocating a signal bit for your port.  See the "Input and
       Output Methods" chapter of the Intuition Reference Manual for full
        details
     - if the IDCMP for the given window exists, and the
       IDCMPFlags argument is zero, this says that you want
       Intuition to close the ports, free the buffers and free
       your signal bit.  You MUST be the same task that was active
       when this signal bit was allocated (either by ModifyIDCMP()
        or OpenWindow() ).
     - if the IDCMP for the given window is opened, and the IDCMPFlags
       argument is not zero, this means that you want to change the
       state of which events will be broadcast to you through the IDCMP

       NOTE:  You can set up the Window->UserPort to any port of your own
       before you call ModifyIDCMP().  If IDCMPFlags is non-null but
       your UserPort is already initialized, Intuition will assume that
       it's a valid port with task and signal data preset and Intuition
       won't disturb your set-up at all, Intuition will just allocate
       the Intuition message port half of it.  The converse is true
       as well:  if UserPort is NULL when you call here with
       IDCMPFlags == NULL, Intuition will deallocate only the Intuition
       side of the port.

       This allows you to use a port that you already have allocated:
     - OpenWindow() with IDCMPFlags equal to NULL (open no ports)
     - set the UserPort variable of your window to any valid port of your
       own choosing
     - call ModifyIDCMP with IDCMPFlags set to what you want
     - then, to clean up later, set UserPort equal to NULL before calling
       CloseWindow() (leave IDCMPFlags alone)  BUT FIRST: you must make
        sure that no messages sent your window are queued at the port,
        since they will be returned to the memory free pool.

       For an example of how to close a window with a shared IDCMP,
       see the description for CloseWindow().


   INPUTS
       Window = pointer to the Window structure containing the IDCMP ports
       IDCMPFlags = the flag bits describing the new desired state of the IDCMP

   RESULT
```

```
    Starting in V37, this function returns NULL if it was unable
    to create the necessary message ports.  (The possibility of
    failure exists in earlier releases, but no return code was offered).
    Do not check the return code under V36 or earlier.

BUGS

SEE ALSO
    OpenWindow(), CloseWindow()
```

```
intuition.library/ModifyProp                         intuition.library/ModifyProp

   NAME
       ModifyProp -- Modify the current parameters of a proportional gadget.

   SYNOPSIS
       ModifyProp( Gadget, Window, Requester,
                   A0      A1      A2
                   Flags, HorizPot, VertPot, HorizBody, VertBody )
                   D0     D1        D2       D3         D4

       VOID ModifyProp( struct Gadget *, struct Window *,
               struct Requester *, UWORD, UWORD, UWORD, UWORD, UWORD );

   FUNCTION
       Modifies the parameters of the specified proportional gadget.  The
       gadget's internal state is then recalculated and the imagery
       is redisplayed in the window or requester that contains the gadget.

       The requester variable can point to a requester structure.  If the
       gadget has the GTYP_REQGADGET flag set, the gadget is in a requester
       and the window pointer must point to the window of the requester.
       If this is not the gadget of a requester, the requester argument may
       be NULL.

       NOTE: this function causes all gadgets from the proportional
       gadget to the end of the gadget list to be refreshed, for
       reasons of compatibility.
       For more refined display updating, use NewModifyProp().

       New for V36: ModifyProp() refreshing consists of redrawing gadgets
       completely.  NewModifyProp() has changed this behavior (see
       NewModifyProp()).

   INPUTS
       PropGadget = pointer to a proportional gadget
       Window = pointer to the window containing the gadget or the window
           containing the requester containing the gadget.
       Requester = pointer to a requester (may be NULL if this isn't
           a requester gadget)
       Flags = value to be stored in the Flags field of the PropInfo
       HorizPot = value to be stored in the HorizPot field of the PropInfo
       VertPot = value to be stored in the VertPot field of the PropInfo
       HorizBody = value to be stored in the HorizBody field of the PropInfo
       VertBody = value to be stored in the VertBody field of the PropInfo

   RESULT
       None

   BUGS

   SEE ALSO
       NewModifyProp()
       The Intuition Reference Manual and Amiga Rom Kernel Manual contain
       more information on Proportional Gadgets.
```

```
intuition.library/MoveScreen                       intuition.library/MoveScreen

    NAME
        MoveScreen -- Attempt to move the screen by the increments provided.

    SYNOPSIS
        MoveScreen( Screen, DeltaX, DeltaY )
                    A0      D0      D1

        VOID MoveScreen( struct Screen *, WORD, WORD );

    FUNCTION
        Moves the screen the specified increment, specified in screen
        pixel resolution coordinates.

        New for V36: Screen movement limits have been greatly relaxed,
        to support screen scrolling.  In particular, negative values
        for screen LeftEdge and TopEdge may now be valid.

       If the DeltaX and DeltaY variables you specify would move the screen
       in a way that violates any restrictions, the screen will be moved
       as far as possible.  You may examine the LeftEdge and TopEdge fields
        of the Screen structure after this function returns to see where
        the screen really ended up.

        In operation, this function determines what the resulting position
        values that are actually to be used, sets these up, and calls
        MakeScreen() and RethinkDisplay().

    INPUTS
        Screen = pointer to a Screen structure
        DeltaX = amount to move the screen on the x-axis
                Note that DeltaX no longer (V36) need be set to zero
        DeltaY = amount to move the screen on the y-axis
                Note that these coordinates are in the same resolution
                as the screen (such as HIRES or INTERLACE)

    RESULT
        None

    BUGS

    SEE ALSO
        RethinkDisplay()
```

```
intuition.library/MoveWindow                       intuition.library/MoveWindow

    NAME
        MoveWindow -- Ask Intuition to move a window.

    SYNOPSIS
        MoveWindow( Window, DeltaX, DeltaY )
                    A0      D0      D1

        VOID MoveWindow( struct Window *, WORD, WORD );

    FUNCTION
        This routine sends a request to Intuition asking to move the window
        the specified distance.  The delta arguments describe how far to
        move the window along the respective axes.

        Note that the window will not be moved immediately, but rather
        will be moved the next time Intuition receives an input event,
        which happens currently at a minimum rate of ten times per second,
        and a maximum of sixty times a second.

        Interactions with other arbitration of Intuition data structures
        may defer this operation longer.  For V36, you can use the new
        IDCMP class IDCMP_CHANGEWINDOW to detect when this operation has
        completed.

        New for V36: Intuition now will do validity checking on the final
        position.  To send absolute movements, or to move and size a
        window in one step, use ChangeWindowBox().

    INPUTS
        Window = pointer to the structure of the Window to be moved
        DeltaX = how far to move the Window on the x-axis
        DeltaY = how far to move the Window on the y-axis

    RESULT
        None

    BUGS

    SEE ALSO
        ChangeWindowBox(), SizeWindow(), WindowToFront(), WindowToBack()
```

```
intuition.library/MoveWindowInFrontOf   intuition.library/MoveWindowInFrontOf

    NAME
        MoveWindowInFrontOf -- Arrange the relative depth of a window. (V36)

    SYNOPSIS
        MoveWindowInFrontOf( Window, BehindWindow )
                             A0      A1

        VOID MoveWindowInFrontOf( struct Window *, struct Window * );

    FUNCTION
        Depth-arranges a window in front of an another window.
        Brings out the layers.library MoveLayerInFrontOf() to the
        Intuition user.

    INPUTS
        Window =  window to re-position in front of another window
        BehindWindow =  window to re-position in front of

    RESULT
        Repositions window.

    BUGS
        Doesn't respect backdrop windows.

    SEE ALSO
        WindowToFront(), WindowToBack(), layers.library/MoveLayerInFrontOf()
```

```
intuition.library/NewModifyProp             intuition.library/NewModifyProp

    NAME
        NewModifyProp -- ModifyProp(), but with selective refresh.

    SYNOPSIS
        NewModifyProp( Gadget, Window, Requester, Flags,
                       A0      A1      A2         D0
                HorizPot, VertPot, HorizBody, VertBody, NumGad )
                D1        D2       D3         D4        D5

        VOID NewModifyProp( struct Gadget *, struct Window *,
                struct Requester *, UWORD, UWORD, UWORD, UWORD, UWORD, WORD );

    FUNCTION
        Performs the function of ModifyProp(), but refreshes
        gadgets in the list as specified by the NumGad parameter.
        With NumGad = -1, this function is identical to ModifyProp().

        New for V36: When NumGad = 1, this function will now perform
        an incremental update of the proportional gadget knob image,
        rather than refreshing the entire gadget.  This means much
        less flashing when programmatically scrolling a proportional
        gadget.

    INPUTS
        PropGadget = pointer to a proportional gadget
        Window = pointer to the window containing the gadget or the window
            containing the requester containing the gadget.
        Requester = pointer to a requester (may be NULL if this isn't
            a requester gadget)
        Flags = value to be stored in the Flags field of the PropInfo
        HorizPot = value to be stored in the HorizPot field of the PropInfo
        VertPot = value to be stored in the VertPot field of the PropInfo
        HorizBody = value to be stored in the HorizBody field of the PropInfo
        VertBody = value to be stored in the VertBody field of the PropInfo
        NumGad = number of gadgets to be refreshed after propgadget internals
            have been adjusted.  -1 means "to end of list."

    RESULT
        None

    BUGS

    SEE ALSO
        ModifyProp()
        The Intuition Reference Manual contains more information on
        Proportional Gadgets.
```

```
intuition.library/NewObject                       intuition.library/NewObject

    NAME
        NewObjectA -- Create an object from a class. (V36)
        NewObject -- Varargs stub for NewObjectA(). (V36)

    SYNOPSIS
        object = NewObjectA( class, classID, tagList )
        D0                   A0     A1       A2

        APTR NewObjectA( struct IClass *, UBYTE *, struct TagItem * );

        object = NewObject( class, classID, Tag1, ... )

        APTR NewObject( struct IClass *, UBYTE *, ULONG, ... );

    FUNCTION
        This is the general method of creating objects from 'boopsi' classes.
        ('Boopsi' stands for "basic object-oriented programming system for
        Intuition".)

        You specify a class either as a pointer (for a private class) or
        by its ID string (for public classes).  If the class pointer
        is NULL, then the classID is used.

        You further specify initial "create-time" attributes for the
        object via a TagItem list, and they are applied to the resulting
        generic data object that is returned.  The attributes, their meanings,
        attributes applied only at create-time, and required attributes
        are all defined and documented on a class-by-class basis.

    INPUTS
        class = abstract pointer to a boopsi class gotten via MakeClass().
        classID = the name/ID string of a public class.  This parameter is
                only used if 'class' is NULL.
        tagList = pointer to array of TagItems containing attribute/value
                pairs to be applied to the object being created

    RESULT
        A boopsi object, which may be used in different contexts such
        as a gadget or image, and may be manipulated by generic functions.
        You eventually free the object using DisposeObject().

    NOTES
        This function invokes the OM_NEW "method" for the class specified.

    BUGS
        Typedef's for 'Object' and 'Class' are defined in the include
        files but not used consistently.  The generic type APTR is
        probably best used for object and class "handles", with the
        type (UBYTE *) used for classID strings.

    SEE ALSO
        DisposeObject(), SetAttrs(), GetAttr(), MakeClass(),
        Document "Basic Object-Oriented Programming System for Intuition"
        and the "boopsi Class Reference" document.
```

```
intuition.library/NextObject                     intuition.library/NextObject

    NAME
        NextObject -- iterate through the object on an Exec list. (V36)

    SYNOPSIS
        object = NextObject( objectPtrPtr )
        D0                   A0

        APTR NextObject( APTR );

    FUNCTION
        This function is for boopsi class implementors only.

        When you collect a set of boopsi objects on an Exec List
        structure by invoking their OM_ADDMEMBER method, you
        can (only) retrieve them by iterations of this function.

        Works even if you remove and dispose the returned list
        members in turn.

    INPUTS
        Initially, you set a pointer variable to equal the
        lh_Head field of the list (or mlh_Head field of a MinList).
        You pass the *address* of that pointer repeatedly
        to NextObject() until it returns NULL.

    EXAMPLE

        /* here is the OM_DISPOSE case of some class's dispatcher */
        case OM_DISPOSE:
            /* dispose members  */
            object_state = mydata->md_CollectionList.lh_Head;
            while ( member_object = NextObject( &object_state ) )
            {
                DoMethod( member_object, OM_REMOVE ); /* remove from list */
                DM( member, msg );      /* and pass along dispose      */
            }

    RESULT
        Returns pointers to each object in the list in turn, and NULL
        when there are no more.

    NOTES

    BUGS

    SEE ALSO
        DisposeObject(), SetAttrs(), GetAttr(), MakeClass(),
        Document "Basic Object-Oriented Programming System for Intuition"
        and the "boopsi Class Reference" document.
```

```
intuition.library/NextPubScreen              intuition.library/NextPubScreen

    NAME
        NextPubScreen -- Identify next public screen in the cycle. (V36)

    SYNOPSIS
        Buff = NextPubScreen( Screen, NameBuff )
        D0                    A0      A1

        UBYTE *NextPubScreen( struct Screen *, UBYTE * );

    FUNCTION
        Returns name of next public screen in system rotation, to
        allow visitor windows to provide function to "jump" among
        public-screens in a cycle.

    INPUTS
        Screen = pointer to the screen your window is currently open in,
          or NULL, if you don't have a pointer to a public screen.
        NameBuff = pointer to a buffer of MAXPUBSCREENNAME characters,
          for Intuition to fill in with the name of the next public
          screen in rotation.

    RESULT
        Returns NULL if there are no public screens, otherwise a
        pointer to your NameBuff.

    NOTES
        There is no guarantee that the public screen whose name
        was returned by this function will exist or be in "public" state
        by the time you call LockPubScreen(), etc.  You must handle
        cases where LockPubScreen(), etc. will fail.

    BUGS
        The starting screen and cycle order of the public screens isn't
        defined, so do not draw conclusions about the order you
        see in the current version of Intuition.  We reserve the
        right to add meaning to the ordering at a future time.

    SEE ALSO
        OpenScreen(), Intuition V36 update documentation
```

```
intuition.library/ObtainGIRPort              intuition.library/ObtainGIRPort

    NAME
        ObtainGIRPort -- Set up a RastPort for a custom gadget. (V36)

    SYNOPSIS
        RPort = ObtainGIRPort( GInfo )
        D0                     A0

        struct RastPort *ObtainGIRPort( struct GadgetInfo * );

    FUNCTION
        Sets up a RastPort for use (only) by custom gadget hook routines.
        This function must be called EACH time a hook routine needing
        to perform gadget rendering is called, and must be accompanied
        by a corresponding call to ReleaseGIRPort().

        Note that if a hook function passes you a RastPort pointer,
        e.g., GM_RENDER, you needn't call ObtainGIRPort() in that case.

    INPUTS
        A pointer to a GadgetInfo structure, as passed to each custom
        gadget hook function.

    RESULT
        A pointer to a RastPort that may be used for gadget rendering.
        This pointer may be NULL, in which case you should do no rendering.
        You may (optionally) pass a null return value to ReleaseGIRPort().

    BUGS

    SEE ALSO
        ReleaseGIRPort(), Custom Gadget documentation
```

```
intuition.library/OffGadget                       intuition.library/OffGadget

    NAME
        OffGadget -- Disable the specified gadget.

    SYNOPSIS
        OffGadget( Gadget, Window, Requester )
                   A0      A1      A2

        VOID OffGadget( struct Gadget *, struct Window *,
                struct Requester * );

    FUNCTION
        This command disables the specified gadget.  When a gadget is
        disabled, these things happen:
              - its imagery is displayed ghosted
              - the GFLG_DISABLED flag is set
              - the gadget cannot be selected by User

        The window parameter must point to the window which contains the
        gadget, or which contains the requester that contains the gadget.
        The requester parameter must only be valid if the gadget has the
        GTYP_REQGADGET flag set, a requirement for all requester gadgets.

       NOTE:  it's never safe to tinker with the gadget list yourself.  Don't
       supply some gadget that Intuition hasn't already processed in
       the usual way.

        NOTE: for compatibility reasons, this function will refresh all
        gadgets in a requester, and all gadgets from gadget to the
        end of the gadget list if gadget is in a window.

        If you want to improve on this behavior, you may perform the
        equivalent operation yourself: remove a gadget or gadgets,
        change the state of their GFLG_DISABLED flag, replace the
        gadgets using AddGList(), and selectively call RefreshGList().

    INPUTS
        Gadget = pointer to the gadget that you want disabled
        Window = pointer to a window structure containing the gadget or
            containing the requester which contains the gadget
        Requester = pointer to a requester (may by NULL if this isn't
           a requester gadget (i.e. GTYP_REQGADGET is not set)).

    RESULT
        None

    BUGS

    SEE ALSO
        AddGadget(), RefreshGadgets()
```

```
intuition.library/OffMenu                           intuition.library/OffMenu

    NAME
        OffMenu -- Disable the given menu or menu item.

    SYNOPSIS
        OffMenu( Window, MenuNumber )
                 A0      D0

        VOID OffMenu( struct Window *, UWORD );

    FUNCTION
        This command disables a sub-item, an item, or a whole menu.
        This depends on the contents of the data packed into MenuNumber,
        which is described in the Intuition Reference Manual.

    INPUTS
        Window = pointer to the window
        MenuNumber = the menu piece to be disabled

    RESULT
        None

    BUGS

    SEE ALSO
        OnMenu(), ResetMenuStrip()
```

```
intuition.library/OnGadget                        intuition.library/OnGadget

    NAME
        OnGadget -- Enable the specified gadget.

    SYNOPSIS
        OnGadget( Gadget, Window, Requester )
                  A0      A1      A2

        VOID OnGadget( struct Gadget *, struct Window *,
                struct Requester * );

    FUNCTION
        This command enables the specified gadget.  When a gadget is
        enabled, these things happen:
        - its imagery is displayed normally (not ghosted)
        - the GFLG_DISABLED flag is cleared
        - the gadget can thereafter be selected by the user

        The window parameter must point to the window which contains the
        gadget, or which contains the requester that contains the gadget
        The requester parameter must only be valid if the gadget has the
        GTYP_REQGADGET flag set, a requirement for all requester gadgets.

       NOTE:  it's never safe to tinker with the gadget list yourself.  Don't
       supply some gadget that Intuition hasn't already processed in
       the usual way.

        NOTE: for compatibility reasons, this function will refresh all
        gadgets in a requester, and all gadgets from gadget to the
        end of the gadget list if gadget is in a window.

        If you want to improve on this behavior, you may perform the
        equivalent operation yourself: remove a gadget or gadgets,
        change the state of their GFLG_DISABLED flag, replace the
        gadgets using AddGList(), and selectively call RefreshGList().

    INPUTS
        Gadget = pointer to the gadget that you want disabled
        Window = pointer to a window structure containing the gadget or
            containing the requester which contains the gadget
        Requester = pointer to a requester (may by NULL if this isn't
           a requester gadget (i.e. GTYP_REQGADGET is not set)).

    RESULT
        None

    BUGS

    SEE ALSO
```

```
intuition.library/OnMenu                              intuition.library/OnMenu

    NAME
        OnMenu -- Enable the given menu or menu item.

    SYNOPSIS
        OnMenu( Window, MenuNumber )
                A0      D0

        VOID OnMenu( struct Window *, UWORD );

    FUNCTION
        This command enables a sub-item, an item, or a whole menu.
        This depends on the contents of the data packed into MenuNumber,
        which is described in the Intuition Reference Manual.

    INPUTS
        Window = pointer to the window
        MenuNumber = the menu piece to be enables

    RESULT
        None

    BUGS

    SEE ALSO
        OffMenu(), ResetMenuStrip()
```

```
intuition.library/OpenScreen                   intuition.library/OpenScreen

   NAME
       OpenScreen -- Open an Intuition screen.

   SYNOPSIS
       Screen = OpenScreen( NewScreen )
       D0                   A0

       struct Screen *OpenScreen( struct NewScreen * );

       or

       struct Screen *OpenScreen( struct ExtNewScreen * );

   FUNCTION
       Opens an Intuition screen according to the specified parameters
       found in the NewScreen structure.

       Does all the allocations, sets up the screen structure and all
       substructures completely, and links this screen's viewport into
       Intuition's View structure.

       Before you call OpenScreen(), you must initialize an instance of
       a NewScreen structure.  NewScreen is a structure that contains
       all of the arguments needed to open a screen.  The NewScreen
       structure may be discarded immediately after OpenScreen() returns.

       The SHOWTITLE flag is set to TRUE by default when a screen is opened.
       To change this, you must call the routine ShowTitle().

   INPUTS
       NewScreen = pointer to an instance of a NewScreen structure.

       New for V36:
       In addition to the information contained in the NewScreen structure,
       Intuition now recognizes extended data passed in the form
       of an array of TagItem structures (from <utility/tagitem.h>),
       commonly called a "tag list."

       There are two ways to provide this array.  The first is to
       use the new Intuition entry point OpenScreenTagList() and
       pass the tag list as a parameter.  This is the recommended
       method, and has a convenient format variation for C programs
       using a variable number of arguments.

       An older way used for some V36 development uses the OpenScreen()
       entry point, and an extension of the NewScreen structure named
       ExtNewScreen.  See the documentation of the flag NS_EXTENDED,
       below.

       While we recommend that you use OpenScreenTagList() rather than
       OpenScreen() when using the extension tag list, we document
       the tag ID values here, so that all parameters for opening
       a screen can be found in one place.

       NewScreen is initialized with the following information:
       -------------------------------------------------------------
       Left = initial x-position of your screen (should be zero for
       releases prior to V36)

       Top = initial y-position of the opening screen
       (Note: Left and Top are specified relative to the Intuition's view,
       in same resolution as the screen pixels.)

       Width = the width for this screen's RastPort
```

```
       Height = the height for his screen's RastPort, or the constant
               STDSCREENHEIGHT to get the current default height (at
               this time guaranteed to be at least 200 rows).  The normal
               width and height for a particular system is stored by
               the graphics.library in GfxBase->NormalDisplayRows and
               GfxBase->NormalDisplayColumns.  These values will be different
               depending on factors such as PAL video and overscan.

               For V36, a new constant STDSCREENWIDTH is introduced.  It
               serves the similar function for screen width.  Both
               STDSCREENWIDTH and STDSCREENHEIGHT indicate that your
               screen RastPort is to be the same dimensions as your
               DisplayClip rectangle.  If you do not specify either a
               standard or custom DisplayClip, the OSCAN_TEXT region
               will be used, which corresponds to the standard dimensions
               of V35 and earlier.

               Furthermore, if you are using OpenScreenTagList(), and you
               specify STDSCREENWIDTH, and you DO NOT provide a NewScreen
               pointer, and you DO NOT provide SA_Left, then Intuition
               will automatically set the LeftEdge of the screen to
               be the left edge of the screen's DisplayClip region.
               Likewise for STDSCREENHEIGHT and the screen's TopEdge.

       Depth = number of bitplanes

       DetailPen = pen number for details (like gadgets or text in title bar)
           The common value for this pen is 0.

       BlockPen = pen number for block fills (like title bar)
           The common value for this pen is 1.

       Type = screen type values
           Set these flags as desired from the set:
           CUSTOMSCREEN -- this is your own screen, not a system screen.
           CUSTOMBITMAP -- this custom screen has bit maps supplied
               in the bitmap field of the NewScreen structure.  Intuition is
               not to allocate any raster bitmaps.
           SCREENBEHIND -- your screen will be created behind all other open
               screens.  This allows a program to prepare imagery in the
               screen, change its colors, and so on, bringing it to the
               front when it is presentable.
           SCREENQUIET -- Intuition will not render system screen gadgets or
               screen title.  In concert with the WFLG_RMBTRAP flag on all
               your screen's windows, this flag will prevent Intuition from
               rendering into your screen's bitplanes.  Without WFLG_RMBTRAP
               (or using the IDCMP_MENUVERIFY facility to cancel menu
               operations), this flag will prevent Intuition from clearing
               your menu bar, which is probably unacceptable.  The menu bar
               layer may still overwrite  a portion of your screen bitmap
               when the screen is opened.  (V36: it won't clobber your bits
               any more.)
           NS_EXTENDED for this screen to use extended attributes pointed
               to by the 'Extended' field, below.

       ViewModes = the appropriate argument for the data type ViewPort.Modes.
          These include:
          HIRES for this screen to be HIRES width.
          INTERLACE for the display to switch to interlace.
          SPRITES for this screen to use sprites (the pointer
               sprite is always displayed)
          DUALPF for dual-playfield mode (not supported yet)
           [For V36: The ViewModes field is superceded by a TagItem with
           tag value SA_DisplayID.]
```

```
Font = pointer to the default TextAttr structure for text in this
    screen and all windows that open in this screen.  Text that uses
    this TextAttr includes title bars of both screen and windows,
    string gadgets, and menu titles.  Of course, IntuiText that
    specifies a NULL TextAttr field will use the screen/window default
    fonts.  NOTE: Intuition will *NOT* call OpenDiskFont(), so
    the TextAttr you supply must be in memory.  The ways to ensure
    that are to either use a ROM font (Topaz 8 or 9) or first
    call OpenDiskFont() to load the font, and don't close it
    until after your screen is successfully opened.
    [For V36: this is superceded by SA_Font and SA_SysFont.]

DefaultTitle = pointer to a line of text that will be displayed along
   the screen's title bar.  Null terminated, or just a NULL pointer
   to get no text
    [For V36: superceded by SA_Title.]

Gadgets = This field should be set to NULL, since no user gadgets may
    be attached to a screen with the current versions of Intuition.

CustomBitMap = if you're not supplying a custom bitmap, this value is
   ignored.  However, if you have your own display memory that you
   want used for this screen, the CustomBitMap field should point to
    the BitMap structure that describes your display memory.  See the
   "Screens" chapter and the "Amiga ROM Kernel Manual" for more
   information about bitmaps.
    [For V36: this is superceded by SA_BitMap.]

[ All TagItem extensions below are new for V36.]
Extension = if NS_EXTENDED is set in NewScreen.Type, this pointer
    should point to an array (or chain of arrays) of TagItems,
    as defined in the include file <utility/tagitem.h>.   This
    field is only defined in the structure ExtNewScreen.
    The values to use for TagItem.ti_Tag are defined below.  We
    recommend that V36-specific applications use the new Intuition
    entry point OpenScreenTagList(), rather than using this field.
    The ExtNewScreen structure is a convenient way to give V36
    Intuition some information that V34 and earlier Intuition will
    ignore.

    Each TagItem is an optional tagged data structure which identifies
    an additional parameter to OpenScreen().  The applicable tag ID
    values for TagItem.ti_Tag and their corresponding data follow.

    Several of the tag items are alternative (and overriding) versions
    to familiar fields in NewScreen.  They are:

SA_Left
SA_Top
SA_Width
SA_Height
SA_Depth
SA_DetailPen
SA_BlockPen
SA_Title
SA_Font
SA_Type
SA_BitMap (whose existence also implies CUSTOMBITMAP).

    Several tags are Booleans, which means that depending on whether
    their corresponding ti_Data field is zero (FALSE) or non-zero
    (TRUE), they specify Boolean attributes.  The ones corresponding
    to Boolean flags in the NewScreen.Type field are:

SA_ShowTitle
SA_Behind (equiv. to SCREENBEHIND)
```

```
SA_Quiet (equiv. to SCREENQUIET)

    The following tags provide extended information to Intuition
    when creating a screen:

SA_DisplayID: ti_Data is a 32-bit extended display mode ID
    as defined in <graphics/displayinfo.h>

SA_Overscan: ti_Data contains a defined constant specifying
    one of the system standard overscan dimensions appropriate for
    the display mode of the screen.  Used with the Width and
    Height dimensions STDSCREENWIDTH and STDSCREEN, this makes
    it trivial to open an overscanned or standard dimension
    screen.  You may also hand-pick your various dimensions
    for overscanned or other screens, by specifying screen position
    and dimensions explicitly, and by using SA_DClip to explicitly
    specify an overscanned DisplayClip region.

    The values for ti_Data of this tag are as follows:

    OSCAN_TEXT - Text Overscan region.  A region which is completely
    on screen and readable ("text safe").  A preferences data
    setting, this is backward equivalent with the old MoreRows,
    and specifies the DisplayClip and default dimensions of the
    Workbench screen.  This is the default.

    OSCAN_STANDARD - Also a preferences setting, this specifies
    a rectangle whose edges are "just out of view."  This yields
    the most efficient position and dimensions of on-monitor
    presentations, such as games and artwork.

    OSCAN_MAX - This is the largest rectangular region that the
    graphics library can handle "comfortably" for a given mode.
    Screens can smoothly scroll (hardware pan) within this region,
    and any DisplayClip or Screen region within this rectangle
    is also legal.  It is not a preferences item, but reflects
    the limits of the graphics hardware and software.

    OSCAN_VIDEO - This is the largest region that the graphics
    library can display, comfortable or not.  There is no guarantee
    that all smaller rectangles are valid.  This region is
    typically out of sight on any monitor or TV, but provides our
    best shot at "edge-to-edge" video generation.

    Remember, using overscan drastically effects memory use and
    chip memory bandwidth.  Always use the smallest (standard)
    overscan region that works for your application.

SA_DClip: ti_Data is a pointer to a rectangle which explicitly
    defines a DisplayClip region for this screen.  See QueryOverscan()
    for the role of the DisplayClip region.

    Except for overscan display screens, this parameter is
    unnecessary, and specifying a standard value using SA_Overscan
    is normally an easier way to get overscan.

SA_AutoScroll: this is a Boolean tag item, which specifies that
    this screens is to scroll automatically when the mouse pointer
    reaches the edge of the screen.  The operation of this requires
    that the screen dimensions be larger than its DisplayClip
    region.

SA_PubName: If this field is present (and ti_Data is non-NULL),
    it means that the screen is a public screen, and that
    the public screen name string is pointed to by ti_Data.
    Public screens are opened in "PRIVATE" mode and must
    be made public using PubScreenStatus().

SA_Pens: The ti_Data field (if non-NULL) points to a UWORD
    array of pen specification, as defined for struct DrawInfo.
    This array will be used to initialize the screen's
    DrawInfo.dri_Pens array.

    SA_Pens is also used to decide that a screen is ready
    to support the full-blown "new look" graphics.  If you
    want the 3D embossed look, you must provide this tag,
    and the ti_Data value cannot be NULL.  If it points
    to a "minimal" array, containing just the terminator ~0,
    you can specify "new look" without providing any values
    for the pen array.

The following two tag items specify the task and signal to be issued
to notify when the last "visitor" window closes on a public screen.
This support is to assist envisioned public screen manager programs.

SA_PubTask:  Task to be signalled.  If absent (and SA_PubSig is
    valid), use the task which called OpenScreen() or
    OpenScreenTagList()).

SA_PubSig:  Data is a UBYTE signal number (not flag) used to notify
    a task when the last visitor window closes on a public screen.

SA_Colors: ti_Data points to an array of ColorSpec structures
    (terminated with ColorIndex = -1) which specify initial
    values of the screen's color palette.

SA_FullPalette: this is a Boolean attribute.  Prior to V36, there
    were just 7 RGB color values that Intuition maintained
    in its user preferences (playfield colors 0-3, and colors
    17-19 for the sprite).  When opening a screen, the color
    map for the screens viewport is first initialized by
    graphics (graphics.library/GetColorMap()) then these
    seven values are overridden to take the preferences values.

    In V36, Intuition maintains a full set of 32 preferences colors.
    If you specify TRUE for SA_FullPalette, Intuition will
    override ALL color map entries with its full suite of
    preferred colors.

SA_ErrorCode: ti_Data points to a ULONG in which Intuition will
    stick an extended error code if OpenScreen[TagList]() fails.
    Values are of this include 0, for success, and:
    OSERR_NOMONITOR     - monitor for display mode not available.
    OSERR_NOCHIPS       - you need newer custom chips for display mode.
    OSERR_NOMEM         - couldn't get normal memory
    OSERR_NOCHIPMEM     - couldn't get chip memory
    OSERR_PUBNOTUNIQUE  - public screen name already used
    OSERR_UNKNOWNMODE   - don't recognize display mode requested

    NOTE: These values are not the same as some similar return
    values defined in graphics.library/ModeNotAvailable().

SA_SysFont: ti_Data selects one of the system standard fonts
    specified in preferences.  This tag item overrides the
    NewScreen.Font field and the SA_Font tag item.

    Values recognized in ti_Data at present are:
        0 - old DefaultFont, fixed-width, the default.
        1 - Workbench screen preferred font.  You have to
            be very font sensitive to handle a proportional or
            larger than traditional screen font.
```

```
        NOTE WELL: if you select sysfont 1, windows opened on
        your screen will not inherit the screen font, but rather
        the window RastPort will be initialized to the old-style
        DefaultFont (sysfont 0).

RESULT
    If all is well, returns the pointer to your new screen
    If anything goes wrong, returns NULL, with further error
    specification in the variable pointed to by the SA_ErrorCode
    data field (V36 and later).

NOTE
    By default, AmigaDOS requesters related to your process are put on
    the Workbench screen (these are messages like "Disk Full").  If
    you wish them to show up on custom screens, DOS must be told.
    This fragment shows the procedure.  More information is available
    in the AmigaDOS manuals.  Sample code fragment:

    #include "libraries/dosextens.h"
            ...
    struct Process   *process;
    struct Window    *window;
    APTR             temp;
            ...
            process = (struct Process *) FindTask(NULL);
            temp = process->pr_WindowPtr;   (save old value)
            process->pr_WindowPtr = (APTR) window;
            ( use a pointer to any open window on your screen )
                    ...
            your code goes here
                    ...
            process->pr_WindowPtr = temp;
            ( restore value BEFORE CloseWindow() )
            CloseWindow(window);

BUGS

SEE ALSO
    OpenScreenTagList(), OpenWindow(), PrintIText(), CloseScreen(),
    QueryOverscan() PubScreenStatus(), The Intuition Reference Manual,
    utility/tagitem.h, graphics.library/ModeNotAvailable(),
    diskfont.library/OpenDiskFont(), graphics.library/GetColorMap()
```

```
intuition.library/OpenScreenTagList       intuition.library/OpenScreenTagList

   NAME
       OpenScreenTagList -- OpenScreen() with TagItem extension array. (V36)
       OpenScreenTags -- Varargs stub for OpenScreenTagList. (V36)

   SYNOPSIS
       Screen = OpenScreenTagList( NewScreen, TagItems )
       D0                          A0         A1

       struct Screen *OpenScreenTagList( struct NewScreen *,
               struct TagItem * );

       Screen = OpenScreenTags( NewScreen, Tag1, ... )

       struct Screen *OpenScreenTags( struct NewScreen *,
               ULONG, ... );

   FUNCTION
       Provides an extension to the parameters passed to OpenScreen().
       This extensions is in the form of (a pointer to) an array of
       TagItem structures, which have to fields: ti_Tag, an ID identifying
       the meaning of the other field, ti_Data.  See <utility/tagitem.h>.

       The tag items can supplement or override the values in NewScreen.
       In fact, you can pass a NULL value of the NewScreen pointer.
       For that matter, if you pass NULL in both arguments, you'll get
       a screen with defaults in all fields, including display mode,
       depth, colors, dimension, title, and so on.  We ask that
       you at least supply a title when you open a screen.

       See OpenScreen() documentation for parameter specifications.

   INPUTS
       NewScreen     - (optional) pointer to a NewScreen structure.
       TagItems      - (optional) pointer to (an array of) TagItem
                         structures, terminated by the value TAG_END.

   RESULT
       Screen  - an open Intuition screen.  See OpenScreen() for
               extended error codes when Screen is returned NULL.

   EXAMPLE
       The version using a variable number of arguments must be
       created for each particular compiler, and may not have
       an analogue in all versions.  For vanilla, 32-bit C
       parameter passing conventions, this works (and will
       appear in amiga.lib):

       struct Screen   *
       OpenScreenTags( ns, tag1 )
       struct NewScreen        *ns;
       ULONG                   tag1;
       {
           struct Screen       *OpenScreenTagList();

           return ( OpenScreenTagList( ns, (struct TagItem *) &tag1 ) );
       }

   NOTES
       We recommend this extension to OpenScreen() over using the
       field ExtNewScreen.Extension.  However, the ExtNewScreen.Extension
       is a convenient way to supply a few tags to V36 Intuition which
       will be ignored by V34 Intuition.  See OpenScreen() documentation
       for lots of details.
```

```
   BUGS

   SEE ALSO
       OpenScreen()
```

```
intuition.library/OpenWindow                        intuition.library/OpenWindow

    NAME
        OpenWindow -- Open an Intuition window.

    SYNOPSIS
        Window = OpenWindow( NewWindow )
        D0                   A0

        struct Window *OpenWindow( struct NewWindow * );

    FUNCTION
        Opens an Intuition window of the given dimensions and position,
        with the properties specified in the NewWindow structure.
        Allocates everything you need to get going.

        New for V36: there is an extensive discussion of public Screens
        and visitor windows at the end of this section.  Also,
        you can provide extensions to the NewWindow parameters using
        and array of TagItem structures.  See the discussion below,
        and the documentation for the function OpenScreenTagList().

        Before you call OpenWindow(), you must initialize an instance of
        a NewWindow structure.  NewWindow is a structure that contains
        all of the arguments needed to open a window.  The NewWindow
        structure may be discarded immediately after it is used to open
        the window.

        If Type == CUSTOMSCREEN, you must have opened your own screen
        already via a call to OpenScreen().  Then Intuition uses your screen
        argument for the pertinent information needed to get your window
        going.  On the other hand, if type == one of the Intuition's standard
        screens, your screen argument is ignored.  Instead,
        Intuition will check to see whether or not that screen
        already exists:  if it doesn't, it will be opened first before
        Intuition opens your window in the standard screen.

        New for V36: If you specify Type == WBENCHSCREEN, then your
        window will appear on the Workbench screen, unless the global
        public screen mode SHANGHAI is set, in which case your window
        will be "hijacked" to the default public screen.  See also
        SetPubScreenModes().

        New for V36: If the WFLG_NW_EXTENDED flag is set, it means that the
        field 'ExtNewWindow->Extension' points to an array of TagItems, as
        defined in intuition/tagitem.h.  This provides an extensible means
        of providing extra parameters to OpenWindow.  For compatibility
        reasons, we could not add the 'Extension' field to the NewWindow
        structure, so we have define a new structure ExtNewWindow, which
        is identical to NewWindow with the addition of the Extension field.

        We recommend that rather than using ExtNewWindow.Extension, you
        use the new Intuition function OpenWindowTagList() and its
        varargs equivalent OpenWindowTags().  We document the window
        attribute tag ID's (ti_Tag values) here, rather than in
        OpenWindowTagList(), so that you can find all the parameters
        for a new window defined in one place.

        If the WFLG_SUPER_BITMAP flag is set, the bitmap variable must point
        to your own bitmap.

        The DetailPen and the BlockPen are used for system rendering; for
        instance, the title bar is first filled using the BlockPen, and then
        the gadgets and text are rendered using DetailPen.  You can either
        choose to supply special pens for your window, or, by setting either
        of these arguments to -1, the screen's pens will be used instead.
```

```
        Note for V36: The DetailPen and BlockPen no longer determine
        what colors will be used for window borders, if your window
        opens on a "full-blown new look screen."

    INPUTS
        NewWindow = pointer to an instance of a NewWindow structure.  That
               structure is initialized with the following data:
   -------------------------------------------------------------------------
        Left = the initial x-position for your window
        Top = the initial y-position for your window
        Width = the initial width of this window
        Height = the initial height of this window

        DetailPen = pen number (or -1) for the rendering of window details
         (like gadgets or text in title bar)
        BlockPen = pen number (or -1) for window block fills (like title bar)
        [For V36: Title bar colors are determined otherwise.]

        Flags = specifiers for your requirements of this window, including:
             which system gadgets you want attached to your window:

            - WFLG_DRAGBAR allows this window to be dragged
            - WFLG_DEPTHGADGET lets the user depth-arrange this window
            - WFLG_CLOSEGADGET attaches the standard close gadget
            - WFLG_SIZEGADGET allows this window to be sized.

            If you ask for the WFLG_SIZEGADGET gadget, you must specify one or
            both of the flags WFLG_SIZEBRIGHT and WFLG_SIZEBBOTTOM below; if
            you don't, the default is WFLG_SIZEBRIGHT.  See the following items
            WFLG_SIZEBRIGHT and WFLG_SIZEBBOTTOM for more details.

            - WFLG_SIZEBRIGHT is a special system gadget flag that
              you set to specify whether or not you want the
              RIGHT border adjusted to account for the physical size
              of the sizing gadget.  The sizing gadget must, after
              all, take up room in either the right or bottom border
              (or both, if you like) of the window.  Setting either
              this or the WFLG_SIZEBBOTTOM flag selects which edge
              will take up the slack.  This will be particularly
              useful to applications that want to use the extra space
              for other gadgets (like a proportional gadget and two
              Booleans done up to look like scroll bars) or, for
              for instance, applications that want every possible
              horizontal bit and are willing to lose lines vertically.
              NOTE:  if you select WFLG_SIZEGADGET, you must select
              either WFLG_SIZEBRIGHT or WFLG_SIZEBBOTTOM or both.  If
              you select neither, the default is WFLG_SIZEBRIGHT.
            - WFLG_SIZEBBOTTOM is a special system gadget flag that
              you set to specify whether or not you want the
              BOTTOM border adjusted to account for the physical size
              of the sizing gadget.  For details, refer to
              WFLG_SIZEBRIGHT above.

            - WFLG_GIMMEZEROZERO for easy but expensive output

            what type of window layer you want, either:
            - WFLG_SIMPLE_REFRESH
            - WFLG_SMART_REFRESH
            - WFLG_SUPER_BITMAP

            - WFLG_BACKDROP for whether or not you want this window to be one
              of Intuition's special backdrop windows.  See WFLG_BORDERLESS
              as well.

            - WFLG_REPORTMOUSE for whether or not you want to "listen" to
```

mouse movement events whenever your window is the active one. After you've opened your window, if you want to change you can later change the status of this via a call to ReportMouse(). Whether or not your window is listening to mouse is affected by gadgets too, since they can cause you to start getting reports too if you like. The mouse move reports (either InputEvents or messages on the IDCMP) that you get will have the x/y coordinates of the current mouse position, relative to the upper-left corner of your window (WFLG_GIMMEZEROZERO notwithstanding). This flag can work in conjunction with the IDCMP Flag called IDCMP_MOUSEMOVE, which allows you to listen via the IDCMP.

- WFLG_BORDERLESS should be set if you want a window with no border padding. Your window may have the border variables set anyway, depending on what gadgetry you've requested for the window, but you won't get the standard border lines and spacing that comes with typical windows.

  This is a good way to take over the entire screen, since you can have a window cover the entire width of the screen using this flag. This will work particularly well in conjunction with the WFLG_BACKDROP flag (see above), since it allows you to open a window that fills the ENTIRE screen. NOTE: this is not a flag that you want to set casually, since it may cause visual confusion on the screen. The window borders are the only dependable visual division between various windows and the background screen. Taking away that border takes away that visual cue, so make sure that your design doesn't need it at all before you proceed.

- WFLG_ACTIVATE is the flag you set if you want this window to automatically become the active window. The active window is the one that receives input from the keyboard and mouse. It's usually a good idea to to have the window you open when your application first starts up be an ACTIVATED one, but all others opened later not be ACTIVATED (if the user is off doing something with another screen, for instance, your new window will change where the input is going, which would have the effect of yanking the input rug from under the user). Please use this flag thoughtfully and carefully.

  Some notes: First, your window may or may not be active by the time this function returns. Use the IDCMP_ACTIVEWINDOW IDCMP message to know when your window has become active. Also, be very careful not to mistakenly specify the obsolete flag names WINDOWACTIVE or ACTIVEWINDOW. These are used in other contexts, and their values unintentionally added to your flags can cause most unfortunate results. To avoid confusion, they are now know as WFLG_WINDOWACTIVE and IDCMP_ACTIVEWINDOW.

- WFLG_RMBTRAP, when set, causes the right mouse button events to be trapped and broadcast as events. You can receive these events through either the IDCMP or the console.

- WFLG_NOCAREREFRESH indicates that you do not wish to be responsible for calling BeginRefresh() and EndRefresh() when your window has exposed regions (i.e., when the IDCMP_REFRESHWINDOW message would be generated). See also the descriptions of these two functions.

- WFLG_NW_EXTENDED (V36) indicates that NewWindow in fact points to an ExtNewWindow structure, and that the 'Extension' field points to an array of TagItem structures, with

meaning described below.

IDCMPFlags = IDCMP is the acronym for Intuition Direct Communications Message Port. (It's Intuition's sole acronym.) If any of the IDCMP Flags is selected, Intuition will create a pair of message ports and use them for direct communications with the task opening this window (as compared with broadcasting information via the Console device). See the "Input and Output Methods" chapter of the Intuition Reference Manual for complete details.

You request an IDCMP by setting any of these flags. Except for the special VERIFY flags, every other flag you set tells Intuition that if a given event occurs which your program wants to know about, it is to broadcast the details of that event through the IDCMP rather than via the Console device. This allows a program to interface with Intuition directly, rather than going through the Console device.

Many programs have elected to use IDCMP communication exclusively, and not to associate a console with their windows at all. Some operations, such as IDCMP_MENUVERIFY, can ONLY be achieved using IDCMP.

The IDCMP flags you can set are:

- IDCMP_REQVERIFY is the flag which, like IDCMP_SIZEVERIFY and ...

- IDCMP_MENUVERIFY (see immediately below), specifies that you want to make sure that your graphical state is quiescent before something extraordinary happens. In this case, the extraordinary event is that a rectangle of graphical data is about to be blasted into your Window. If you're drawing directly into its screen, you probably will wish to make sure that you've ceased drawing before the user is allowed to bring up the DMRequest you've set up, and the same for when system has a request for the user. Set this flag to ask for that verification step.

- IDCMP_REQCLEAR is the flag you set to hear a message whenever a requester is cleared from your window. If you are using IDCMP_REQVERIFY to arbitrate access to your screen's bitmap, it is safe to start your output once you have heard an IDCMP_REQCLEAR for each IDCMP_REQSET.

- IDCMP_REQSET is a flag that you set to receive a broadcast for each requester that is opened in your window. Compare this with IDCMP_REQCLEAR above. This function is distinct from IDCMP_REQVERIFY. This functions merely tells you that a requester has opened, whereas IDCMP_REQVERIFY requires you to respond before the requester is opened.

- IDCMP_MENUVERIFY is the flag you set to have Intuition stop and wait for you to finish all graphical output to your window before rendering the menus. Menus are currently rendered in the most memory-efficient way, which involves interrupting output to all windows in the screen before the menus are drawn. If you need to finish your graphical output before this happens, you can set this flag to make sure that you do.

- IDCMP_SIZEVERIFY means that you will be doing output to your window which depends on a knowledge of the current size of the window. If the user wants to resize the window, you may want to make sure that any queued output completes before the sizing takes place (critical text, for instance). If this is the case, set this flag. Then, when the user wants to size, Intuition will send you the IDCMP_SIZEVERIFY message and Wait() until you reply that it's OK to proceed with the sizing. NOTE: when we say that Intuition will Wait() until you reply, what we're really saying is that user will WAIT until you reply, which suffers the great negative potential of User-Unfriendliness. So remember: use this flag sparingly, and, as always with any IDCMP Message you receive, reply to it promptly! Then, after user has sized the window, you can find out about it using IDCMP_NEWSIZE.

With all the "VERIFY" functions, it is not save to leave them enabled at any time when your task may not be able to respond for a long period.

It is NEVER safe to call AmigaDOS, directly or indirectly, when a "VERIFY" function is active. If AmigaDOS needs to put up a disk requester for you, your task might end up waiting for the requester to be satisfied, at the same time as Intuition is waiting for your response. The result is a complete machine lockup. USE ModifyIDCMP() TO TURN OFF ANY VERIFY MESSAGES BEFORE CALLING dos.library!!

For V36: If you do not respond to the verification IntuiMessages within the user specified timeout duration, Intuition will abort the operation. This eliminates the threat of these easy deadlocks, but can result in a confused user. Please try hard to continue to avoid "logical deadlocks".

- IDCMP_NEWSIZE is the flag that tells Intuition to send an IDCMP message to you after the user has resized your window. At this point, you could examine the size variables in your window structure to discover the new size of the window. See also the IDCMP_CHANGEWINDOW IDCMP flag.

- IDCMP_REFRESHWINDOW when set will cause a message to be sent whenever your window needs refreshing. This flag makes sense only with WFLG_SIMPLE_REFRESH and WFLG_SMART_REFRESH windows.

- IDCMP_MOUSEBUTTONS will get reports about mouse-button up/down events broadcast to you (Note: only the ones that don't mean something to Intuition. If the user clicks the select button over a gadget, Intuition deals with it and you don't find out about it through here).

- IDCMP_MOUSEMOVE will work only if you've set the WFLG_REPORTMOUSE flag above, or if one of your gadgets has the GACT_FOLLOWMOUSE flag set. Then all mouse movements will be reported here, providing your window is active.

- IDCMP_GADGETDOWN means that when the User "selects" a gadget you've created with the GACT_IMMEDIATE flag set, the fact will be broadcast through the IDCMP.

- IDCMP_GADGETUP means that when the user "releases" a gadget that you've created with the GACT_RELVERIFY flag set, the fact will be broadcast through the IDCMP. This message is only generated if the release is "good", such as releasing the select button over a Boolean gadget, or typing ENTER in a string gadget.

- IDCMP_MENUPICK selects that menu number data will be sent via the IDCMP.

- IDCMP_CLOSEWINDOW means broadcast the IDCMP_CLOSEWINDOW event

through the IDCMP rather than the console.

- IDCMP_RAWKEY selects that all IDCMP_RAWKEY events are transmitted via the IDCMP. Note that these are absolutely RAW keycodes, which you will have to translate before using. Setting this and the MOUSE flags effectively eliminates the need to open a Console device to get input from the keyboard and mouse. Of course, in exchange you lose all of the console features, most notably the "cooking" of input data and the systematic output of text to your window.

- IDCMP_VANILLAKEY is for developers who don't want the hassle of IDCMP_RAWKEYS. This flag will return all the keycodes after translation via the current country-dependent keymap. When you set this flag, you will get IntuiMessages where the Code field has a decoded ANSI character code representing the key struck on the keyboard. Only codes that map to a single character are returned: you can't read such keys as HELP or the function keys with IDCMP_VANILLAKEY.

  NOTE FOR V36: If you have both IDCMP_RAWKEY and IDCMP_VANILLAKEY set, Intuition will send an IDCMP_RAWKEY event for those *downstrokes* which do not map to single-byte characters ("non-vanilla keys). In this way you can easily detect cursor keys, function keys, and the Help key without sacrificing the convenience of IDCMP_VANILLAKEY.

- IDCMP_INTUITICKS gives you simple timer events from Intuition when your window is the active one; it may help you avoid opening and managing the timer device. With this flag set, you will get only one queued-up INTUITICKS message at a time. If Intuition notices that you've been sent an IDCMP_INTUITICKS message and haven't replied to it, another message will not be sent. Intuition receives timer events and considers sending you an IDCMP_INTUITICKS message approximately ten times a second.

- IDCMP_DELTAMOVE gives raw (unscaled) input event delta X/Y values. This is so you can detect mouse motion regardless of screen/window/display boundaries. This works a little strangely: if you set both IDCMP_MOUSEMOVE and IDCMP_DELTAMOVE. IDCMPFlags, you will get IDCMP_MOUSEMOVE messages with delta x/y values in the MouseX and MouseY fields of the IDCMPMessage.

- IDCMP_NEWPREFS indicates you wish to be notified when the system-wide Preferences changes. For V36, there is a new environment mechanism to replace Preferences, which we recommend you consider using instead.

- Set IDCMP_ACTIVEWINDOW and IDCMP_INACTIVEWINDOW to get messages when those events happen to your window. Take care not to confuse this "ACTIVEWINDOW" with the familiar sounding, but totally different "WINDOWACTIVE" flag. These two flags have been supplanted by "IDCMP_ACTIVEWINDOW" and "WFLG_WINDOWACTIVE". Use the new equivalent terms to avoid confusion.

- Set IDCMP_DISKINSERTED or IDCMP_DISKREMOVED to learn when removable disks are inserted or removed, respectively.

- IDCMP_IDCMPUPDATE is a new class for V36 which is used as a channel of communication from custom and boopsi gadgets to your application.

- IDCMP_CHANGEWINDOW is a new class for V36 that will be sent to your window whenever its dimensions or position are changed

```
        by the user or the functions SizeWindow(), MoveWindow(),
        ChangeWindowBox(), or ZipWindow().

      - IDCMP_MENUHELP is new for V37.  If you specify the WA_MenuHelp
        tag when you open your window, then when the user presses the
        HELP key on the keyboard during a menu session, Intuition will
        terminate the menu session and issue this even in place of an
        IDCMP_MENUPICK message.
          - NEVER follow the NextSelect link for MENUHELP messages.
          - You will be able to hear MENUHELP for ghosted menus.
            (This lets you tell the user why the option is ghosted.)
          - Be aware that you can receive a MENUHELP message whose code
            corresponds to a menu header or an item that has sub-items
            (which does not happen for MENUPICK).  The code may also be
            MENUNULL.
          - LIMITATION:  if the user extend-selects some checkmarked
            items with the mouse, then presses MENUHELP, your
            application will only hear the MENUHELP report.  You
            must re-examine the state of your checkmarks when you
            get a MENUHELP.
          - Availability of MENUHELP in V36 is not directly
            controllable.  We apologize...

 Gadgets = the pointer to the first of a linked list of the your own
        Gadgets which you want attached to this Window.  Can be NULL
        if you have no Gadgets of your own

CheckMark = a pointer to an instance of the struct Image where can
        be found the imagery you want used when any of your
        menu items is to be checkmarked.  If you don't want to
        supply your own imagery and you want to just use
        Intuition's own checkmark, set this argument to NULL

Text = a null-terminated line of text to appear on the title bar of
        your window (may be null if you want no text)

Type = the screen type for this window.  If this equal CUSTOMSCREEN,
        you must have already opened a CUSTOMSCREEN (see text above).
        Types available include:
            - WBENCHSCREEN
            - CUSTOMSCREEN
            - PUBLICSCREEN (new for V36, see text below)

Screen = if your type is one of Intuition's standard screens, then
        this argument is ignored.  However, if Type == CUSTOMSCREEN,
        this must point to the structure of your own screen

BitMap = if you have specified WFLG_SUPER_BITMAP as the type of
         refreshing you want for this window, then this value points to a
         instance of the struct bitmap.  However, if the refresh type
         is NOT WFLG_SUPER_BITMAP, this pointer is ignored.

MinWidth, MinHeight, MaxWidth, MaxHeight = the size limits for this
        window.  These must be reasonable values, which is to say that
        the minimums cannot be greater than the current size, nor can
        the maximums be smaller than the current size.  If they are,
        they're ignored.  Any one of these can be initialized to zero,
        which means that that limit will be set to the current
        dimension of that axis.  The limits can be changed after the
        Window is opened by calling the WindowLimits() routine.

        NOTE: ORIGINALLY, we stated that:

        "If you haven't requested the WFLG_SIZEGADGET option, these
        variables are ignored so you don't have to initialize them."
```

```
        It is now clear that a variety of programs take it upon
        themselves to call SizeWindow() (or ChangeWindowBox()) without
        your program's consent or consulting your WFLG_SIZEGADGE
        option.  To protect yourself against the results, we strongly
        urge that if you supply suitable values for these fields even
        if you do not specify WFLG_SIZEGADGET.

        The maximums may be LARGER than the current size, or even
        larger than the current screen.  The maximums should be set to
        the highest value your application can handle.  This allows
        users with larger display devices to take full advantage of
        your software.  If there is no good reason to limit the size,
        then don't.  -1 or ~0 indicates that the maximum size is only
        limited by the size of the window's screen.

        See also the docs on the function WindowLimits() for more
        information.

Extension (New for V36) = a pointer to an array (or chain of arrays)
     of TagItems to specify additional parameters to OpenWindow().
     TagItems in general are described in utility/tagitem.h,
     and the OpenWindow tags are defined in intuition/intuition.h
     and described here.  For items pertaining to Public Screens
     and visitor windows, please see below.

 Here are the TagItem.ti_Tag values that are defined for OpenWindow
 (and OpenWindowTagList()).

 Certain tags simply override equivalent values in NewWindow,
 and allow you to open a window using OpenWindowTagList() without
 having a NewWindow structure at all.  In each case, cast
 the corresponding data to ULONG and put it in ti_Data.

 The compatible tag items include:

 WA_Left
 WA_Top
 WA_Width
 WA_Height
 WA_DetailPen
 WA_BlockPen
 WA_IDCMP
 WA_Flags           - initial values for Flags before looking at other
                      Boolean component Tag values
 WA_Gadgets
 WA_Checkmark
 WA_Title
 WA_CustomScreen - also implies CUSTOMSCREEN property
 WA_SuperBitMap - also implies WFLG_SUPER_BITMAP refresh mode.
 WA_MinWidth
 WA_MinHeight
 WA_MaxWidth
 WA_MaxHeight

 These Boolean tag items are alternatives to the NewWindow.Flags
 Boolean attributes with similar names.

 WA_SizeGadget           - equivalent to WFLG_SIZEGADGET
 WA_DragBar              - equivalent to WFLG_DRAGBAR
 WA_DepthGadget          - equivalent to WFLG_DEPTHGADGET
 WA_CloseGadget          - equivalent to WFLG_CLOSEGADGET
 WA_Backdrop             - equivalent to WFLG_BACKDROP
 WA_ReportMouse          - equivalent to WFLG_REPORTMOUSE
 WA_NoCareRefresh        - equivalent to WFLG_NOCAREREFRESH
 WA_Borderless           - equivalent to WFLG_BORDERLESS
 WA_Activate             - equivalent to WFLG_ACTIVATE
```

```
WA_RMBTrap              - equivalent to WFLG_RMBTRAP
WA_WBenchWindow         - equivalent to WFLG_WBENCHWINDOW
                          (system PRIVATE)
WA_SimpleRefresh        - only specify if TRUE
WA_SmartRefresh         - only specify if TRUE
WA_SizeBRight           - equivalent to WFLG_SIZEBRIGHT
WA_SizeBBottom          - equivalent to WFLG_SIZEBBOTTOM
WA_GimmeZeroZero        - equivalent to WFLG_GIMMEZEROZERO

The following tag items specify new attributes of a window.

WA_ScreenTitle - You can specify the screen title associated
    with your window this way, and avoid a call to SetWindowTitles()
    when your window opens.

WA_AutoAdjust - a Boolean attribute (default FALSE) which says
    that it's OK to move or even shrink the dimensions of
    this window to fit it on the screen, within the dimension
    limits specified by MinWidth and MinHeight.
    Someday, this processing might be sensitive to the
    currently visible portion of the screen the window
    will be opening on, so don't draw too many conclusions
    about the auto-adjust algorithms.

WA_InnerWidth
WA_InnerHeight - You can specify the dimensions of the interior
    region of your window, independent of what the border
    thicknesses will be.  You probably want to specify
    WA_AutoAdjust to allow Intuition to move your window
    or even shrink it so that it is completely on screen.

    Note: using these tags puts some reasonable restrictions
    on the gadgets you can specify as "border" gadgets when
    you open your window.  Since border gadgets determine
    the border dimensions and hence the overall dimensions of
    your window, those dimensions cannot be used calculating
    the position or dimensions of border gadgets.

    Here's the complete list of restrictions:
    - GACT_LEFTBORDER gadgets cannot be GFLG_RELWIDTH if
      WA_InnerWidth is used.
    - GACT_RIGHTBORDER gadgets MUST be GFLG_RELRIGHT if
      WA_InnerWidth is used.
    - GACT_TOPBORDER gadgets cannot be GFLG_RELHEIGHT if
      WA_InnerHeight is used.
    - GACT_BOTTOMBORDER gadgets MUST be GFLG_RELBOTTOM if
      WA_InnerHeight is used.

WA_PubScreenName - This tag item declares that you want your window
    to open as a visitor window on the public screen whose name
    is pointed to by (UBYTE *) ti_Data.

WA_PubScreen - Open as a visitor window on the public screen
    whose address if provided as (struct Screen *) ti_Data.
    To ensure that this screen remains open long enough, you
    must either:
        1) Be the screen's owner
        2) have another window already open on the screen
        3) use LockPubScreen()
    Using exec.library/Forbid() is not sufficient.

    You can provide ti_Data to be NULL (zero), without any
    of the above precautions, to specify the default public screen.

WA_PubScreenFallBack - This Boolean attribute specifies that a
    visitor window should "fall back" to opening on the default
```

```
        public screen if the explicitly specify public screen is not
        available.

    WA_WindowName - this visionary specification of a window
        rendezvous name string is not yet implemented.

    WA_Colors - this equally great idea about associating a palette
        specification with the active window may not ever be implemented.

    WA_Zoom - ti_Data points to an array of four WORD's to be used
        as the initial Left/Top/Width/Height of the "alternate
        Zoom position and dimensions."  The presence of this tag
        item implies that you want a Zoom gadget, even though you
        might not have a sizing gadget.

    WA_MouseQueue
    WA_RptQueue - These two tags specify limits for the number
        of outstanding IntuiMessages that Intuition will send for
        IDCMP_MOUSEMOVE and repeated IDCMP_RAWKEY, respectively.
        You can change the value of the mouse queue limit after
        the window is open using SetMouseQueue(), but there is
        not currently an equivalent function for the repeat-key queue.

    WA_BackFill - ti_Data is a pointer to a Hook structure that
        the Layers library will call when your window needs
        "backfilling."  See layers.library/InstallLayerHook().

NOTES
    Regarding Public Screens, you can specify a window to be a
    "visitor window" on a public screen in one of several ways.
    In each case, you must be sure not to specify a NewWindow
    type of CUSTOMSCREEN.  You should use the value PUBLICSCREEN.

    There are actually several ways you can specify which screen
    you want a visitor window to be opened on:

        1) Specify the name of the public screen WA_PubScreenName,
        or a NULL pointer, in ti_Data.  The name might have been
        provided by the user.  A NULL pointer means to use the
        default public screen.

        If the named screen cannot be found, the default public screen
        will be used if the Boolean attribute WA_PubScreenFallBack
        is TRUE.

        2) Specify a pointer to a public screen using the
        WA_PubScreen tag item.  The WA_PubScreenFallBack
        attribute has no effect.  You can specify the default
        public screen by providing a NULL pointer.

        You can also specify the pointer by setting NewWindow.Type
        to PUBLICSCREEN, and specifying the public screen pointer
        in NewWindow.Screen.  The WA_PubScreen tag item has precedent
        over this technique.

        Unless NULL, the screen pointer provided MUST be a valid
        public screen.  You may ensure this several ways:

        - Be the owner of the screen.
        - Have a window already open on the screen.
        - Use LockPubScreen() to prevent the screen from closing.
        - specifying the WFLG_VISITOR bit in NewWindow.Flags is not
          supported.

        It is anticipated that the last will be the most common method
        of opening public screens because you often want to examine
        properties of the screen your window will be using in order
        to compensate for differences in dimension, depth, and font.

        The standard sequence for this method is as follows:
        LockPubScreen()      - obtain a pointer and a promise
        layout window        - adapt your window to the screen you will use
        OpenWindow()         - using the pointer you specify
        UnlockPubScreen()    - once your window is open, you can let go
                               of the lock on the public screen
        ... normal window even processing ...
        CloseWindow().

    Regarding "service" windows, such as those opened for a system
    requester or file requester associated with a given "client"window.
    These windows should NOT be "visitor" windows.  Open them
    using NewWindow.Type = CUSTOMSCREEN and NewWindow.Screen
    equal to the screen of the client window (window->WScreen).
    You can also use WA_CustomScreen, which has precedence.

    This ensures that the requester service window will be allowed to
    open on the same screen as the client window, even if that
    screen is not a public screen, or has private status.

    This has an implication for service/client protocol: when you
    pass a window pointer to any system requester routine or
    to a routine which creates some other other service window,
    you MUST keep your window open until the client window
    is closed.

    If a requester service will allow a NULL client window, this
    should indicate to open the service window on the default public
    screen (probably Workbench).  The correct way to get a pointer
    to this screen is to call LockPubScreen( NULL ).  In this
    case, you want to open as a visitor window, which means you
    should use either PUBLICSCREEN or WA_PubScreen, described above.
    You should call UnlockPubScreen() after your visitor window is open.

    As of V36, gadgets in the right and bottom border
    (specified with GACT_RIGHTBORDER and GACT_BOTTOMBORDER) only
    contribute to the dimensions of the borders if they are also
    GFLG_RELRIGHT and GFLG_RELBOTTOM, respectively.

RESULT
    If all is well, returns the pointer to your new Window
    If anything goes wrong, returns NULL

BUGS
    When you open a window, Intuition will set the font of
    the window's RastPort to the font of the window's screen.
    This does not work right for GimmeZeroZero windows: the
    BorderRPort RastPort has the font set correctly, but
    Window.RPort is set up with the system default font.
    For compatibility reasons, we won't be fixing this problem.

    Also, there is a compatibility trick going on with the
    default font of your window's RastPort if the screen's
    font is "fancy."  See the SA_SysFont attribute described
    under OpenScreen().

    Unless you arrange otherwise, each window you open will allocate
    a signal for your task from the 16 "user signals."
    If no signal is available, your window will not be able
    to be opened.  In early V36 versions and before, Intuition didn't
    check this condition, but just left you with an unusable port.

SEE ALSO
```

```
    OpenWindowTagList(), OpenScreen(), ModifyIDCMP(), SetWindowTitles(),
     LockPubScreen(), SetDefaultPubScreen(), ZipWindow(),
     layers.library/InstallLayerHook(), SetPubScreenModes()
```

```
intuition.library/OpenWindowTagList       intuition.library/OpenWindowTagList

   NAME
       OpenWindowTagList -- OpenWindow() with TagItem extension. (V36)
       OpenWindowTags -- Varargs stub for OpenWindowTagList (V36)

   SYNOPSIS
       Window = OpenWindowTagList( NewWindow, TagItems )
       D0                          A0         A1

       struct Window *OpenWindowTagList( struct NewWindow *,
               struct TagItem * );

       Window = OpenWindowTags( NewWindow, Tag1, ... )

       struct Window *OpenWindowTags( struct NewWindow *, ULONG, ... );

   FUNCTION
       A variation of OpenWindow() that allow direct specification of
       a TagItem array of extension data.  Recommended over using the
       ExtNewWindow.Extension field.

       If you omit the NewWindow (pass NULL), a set of defaults
       are used, and overridden by the tag items.  Even without
       any tag items at all, a reasonable window opens on the Workbench
       or default public screen.

       See OpenWindow() for all the details.

   INPUTS
       NewWindow - (optional) pointer to a NewWindow structure.
       TagItems - (optional) pointer to TagItem array, with tag
               values as described under the description for
               OpenWindow().

   RESULT
       Window - newly created window, per your specifications.

   EXAMPLE
       See OpenScreenTagList() for an example of how to create
       a "varargs" version of this function for convenient C
       language programming.

   NOTES


   BUGS


   SEE ALSO
       OpenWindow()
```

```
intuition.library/OpenWorkBench                   intuition.library/OpenWorkBench

   NAME
       OpenWorkBench -- Open the Workbench screen.

   SYNOPSIS
       WBScreen = OpenWorkBench()
       D0

       BOOL OpenWorkBench( VOID );

   FUNCTION
       This routine attempts to reopen the Workbench.  The actions taken are:
           - general good stuff and nice things, and then return a non-null
             pointer to the Workbench screen.
           - find that something has gone wrong, and return NULL

       The return value, if not NULL, is indeed the address of the Workbench
       screen, although you should not use it as such.  This is because the
       Workbench may be closed by other programs, which can invalidate
       the address at any time.  We suggest that you regard the return
       value as a BOOL indication that the routine has succeeded, if
       you pay any attention to it at all.

   INPUTS
       None

   RESULT
       non-zero if Workbench screen opened successfully, or was already
               opened
       FALSE if anything went wrong and the Workbench screen isn't out there

   BUGS
       The name of this routine is spelled wrong: it should be
       OpenWorkbench

   SEE ALSO
```

```
intuition.library/PointInImage                 intuition.library/PointInImage

   NAME
       PointInImage -- Tests whether an image "contains" a point. (V36)

   SYNOPSIS
       DoesContain = PointInImage( Point, Image )
       D0                          D0     A0

       BOOL PointInImage( struct Point, struct Image * );

   FUNCTION
       Tests whether a point is properly contained in an image.
       The intention of this is to provide custom gadgets a means
       to delegate "image mask" processing to the Image, where
       it belongs (superceding things like BOOLMASK).  After all,
       a rounded rect image with a drop shadow knows more about
       what points are inside it than anybody else should.

       For traditional Intuition Images, this routine checks if
       the point is in the Image box (LeftEdge/RightEdge/Width/Height).

   INPUTS
       Point   - Two words, X/Y packed into a LONG, with high word
                 containing 'X'.  This is what you get if you pass
                 a Point structure (not a pointer!) using common
                 C language parameter conventions.
       Image - a pointer to a standard or custom Image data object.
           NOTE: If 'Image' is NULL, this function returns TRUE.

   RESULT
       DoesContain - Boolean result of the test.

   EXAMPLE


   NOTES


   BUGS
       Only applies to the first image, does not follow NextImage
       linked list.  This might be preferred.

   SEE ALSO
```

```
intuition.library/PrintIText                       intuition.library/PrintIText

   NAME
       PrintIText -- Print text described by the IntuiText argument.

   SYNOPSIS
       PrintIText( RastPort, IText, LeftOffset, TopOffset )
                   A0        A1     D0          D1

       VOID PrintIText( struct RastPort *, struct IntuiText *, WORD, WORD );

   FUNCTION
       Prints the IntuiText into the specified RastPort.  Sets up the RastPort
       as specified by the IntuiText values, then prints the text into the
       RastPort at the IntuiText x/y coordinates offset by the left/top
       arguments.  Note, though, that the IntuiText structure itself
       may contain further text position coordinates: those coordinates
       and the Left/TopOffsets are added to obtain the true position of
       the text to be rendered.

       This routine does window layer clipping as appropriate -- if you
       print text outside of your window, your characters will be
       clipped at the window's edge, providing you pass your window's
       (layered) RastPort.

       If the NextText field of the IntuiText argument is non-NULL,
       the next IntuiText is rendered as well, and so on until some
       NextText field is NULL.

       IntuiText with the ITextFont field NULL are displayed in the
       font of the RastPort.  If the RastPort font is also NULL, the
       system default font, as set via the Preferences tool, will be used.

   INPUTS
       RastPort = the RastPort destination of the text
       IText = pointer to an instance of the structure IntuiText
       LeftOffset = left offset of the IntuiText into the RastPort
       TopOffset = top offset of the IntuiText into the RastPort

   RESULT
       None

   BUGS

   SEE ALSO
```

```
intuition.library/PubScreenStatus          intuition.library/PubScreenStatus

    NAME
        PubScreenStatus -- Change status flags for a public screen. (V36)

    SYNOPSIS
        ResultFlags = PubScreenStatus( Screen, StatusFlags )
        D0                             A0      D0

        UWORD PubScreenStatus( struct Screen *, UWORD );

    FUNCTION
        Changes status flags for a given public screen.

        Do not apply this function to a screen if your program
        isn't the screen's "owner", in particular, don't call
        this function for the Workbench screen.

    INPUTS
        Screen = pointer to public screen
        StatusFlags = values currently:
          PSNF_PRIVATE: make this screen unavailable to visitor windows

    RESULT
        Returns 0 in the lowest order bit of the return value
        if the screen wasn't public, or because it can not be taken
        private because visitors are open in it.

        All other bits in the return code are reserved for future
        enhancement.

    BUGS

    SEE ALSO
        OpenScreen(), Intuition V36 update documentation
```

```
intuition.library/QueryOverscan              intuition.library/QueryOverscan

    NAME
        QueryOverscan -- Inquire about a standard overscan region. (V36)

    SYNOPSIS
        success = QueryOverscan( DisplayID, Rect, OScanType )
        D0                       A0         A1    D0

        LONG QueryOverscan( ULONG, struct Rectangle *, WORD );

    FUNCTION
        This function fills in a rectangle with one of the system
        overscan dimensions, scaled appropriately for the mode of
        the DisplayID it is passed.

        There are three types of system overscan values:
        OSCAN_TEXT: completely visible, by user preference.  Used
          for Workbench screen and screen dimensions STDSCREENWIDTH
          and STDSCREENHEIGHT.  Left/Top is always 0,0.
        OSCAN_STANDARD: just beyond visible bounds of monitor, by
          user preference.  Left/Top may be negative.
        OSCAN_MAX: The largest region we can display, AND display
          any smaller region (see note below).
        OSCAN_VIDEO: The absolute largest region that the graphics.library
          can display.  This region must be used as-is.


    INPUTS
        DisplayID -- A 32-bit identifier for a display mode, as
          defined in graphics/displayinfo.h.

          NOTE: If you only intend to use one of the four standard
          overscan dimensions as is, and open your screen to exactly
          the DisplayClip dimensions, you can specify one of
          the OSCAN_ values in the ExtNewScreen tag SA_StdDClip,
          and specify STDSCREENWIDTH and STDSCREENHEIGHT as the
          dimensions to more easily open up an overscanned screen
          (or use no NewScreen in OpenScreenTagList() and accept
          the default standard screen dimensions).

        Rect -- pointer to a Rectangle structure which this function
          will fill out with its return values.  Note that to convert
          a rectangle to a screen "Height" you do (MaxY - MinY + 1), and
          similarly for "Width."  The rectangle may be passed directly
          to OpenScreen() as a DisplayClip region (SA_DClip).

    RESULT
        0 (FALSE) if the MonitorSpec your NewScreen requests
        does not exist.  Non-zero (TRUE) if it does.

    BUGS
        Change in parameter V36.A17 might cause problems for some.

    SEE ALSO
        OpenScreen(), Intuition V36 update documentation
```

```
intuition.library/RefreshGadgets             intuition.library/RefreshGadgets

    NAME
        RefreshGadgets -- Refresh (redraw) the gadget display.

    SYNOPSIS
        RefreshGadgets( Gadgets, Window, Requester )
                        A0       A1      A2

        VOID RefreshGadgets( struct Gadget *, struct Window *,
                struct Requester * );

    FUNCTION
        Refreshes (redraws) all of the gadgets in the gadget list starting
        from the specified gadget.

        The window parameter must point to the window which contains the
        gadget, or which contains the requester that contains the gadget
        The requester parameter must only be valid if the gadget has the
        GTYP_REQGADGET flag set, a requirement for all requester gadgets.

        The Pointer argument points to a Window structure.

        The two main reasons why you might want to use this routine are:
        first, that you've modified the imagery of the gadgets in your
        display and you want the new imagery to be displayed; secondly,
        if you think that some graphic operation you just performed
        trashed the gadgetry of your display, this routine will refresh
        the imagery for you.

        Note that to modify the imagery of a gadget, you must first remove
        that gadget from the window's gadget list, using RemoveGadget() (or
        RemoveGList()).  After changing the image, border, text (including
        text for a string gadget), the gadget is replaced in the gadget list
        (using AddGadget() or AddGList()).  Adding gadgets does not cause
        them to be displayed (refreshed), so this function, or RefreshGList()
        is typically called.

        A common technique is to set or reset the GFLG_SELECTED flag of a
        Boolean gadget and then call RefreshGadgets() to see it displayed
        highlighted if and only if GFLG_SELECTED is set.  If you wish to do
        this and be completely proper, you must RemoveGadget(), change the
        GFLG_SELECTED flag, AddGadget(), and RefreshGadgets(), or the
        equivalent.

        The gadgets argument can be a copy of the FirstGadget variable in
        the Window structure that you want refreshed:
        the effect of this will be that all gadgets will be redrawn.
        However, you can selectively refresh just some of the gadgets
        by starting the refresh part-way into the list:  for instance,
        redrawing your window non-GTYP_GZZGADGET gadgets only, which you've
        conveniently grouped at the end of your gadget list.

        Even more control is available using the RefreshGList() routine which
        enables you to refresh a single gadget, or number of your choice.

        NOTE:  It's never safe to tinker with the gadget list yourself.  Don't
        supply some gadget list that Intuition hasn't already processed in
        the usual way.

    INPUTS
        Gadgets = pointer to the first in the list of gadgets wanting
            refreshment
        Window = pointer to the window containing the gadget or its requester
        Requester = pointer to a requester (ignored if gadget is not attached
            to a requester).
```

```
    RESULT
        None

    BUGS

    SEE ALSO
        RefreshGList(), RemoveGadget(), RemoveGList(), AddGadget(), AddGList()
```

```
intuition.library/RefreshGList                    intuition.library/RefreshGList

    NAME
        RefreshGList -- Refresh (redraw) a chosen number of gadgets.

    SYNOPSIS
        RefreshGList( Gadgets, Window, Requester, NumGad )
                      A0       A1      A2         D0

        VOID RefreshGList( struct Gadget *, struct Window *,
                struct Requester *, WORD );

    FUNCTION
        Refreshes (redraws) gadgets in the gadget list starting
        from the specified gadget.  At most NumGad gadgets are redrawn.
        If NumGad is -1, all gadgets until a terminating NULL value
        in the NextGadget field is found will be refreshed, making this
        routine a superset of RefreshGadgets().

        The Requester parameter can point to a Requester structure.  If
        the first gadget in the list has the GTYP_REQGADGET flag set, the
        gadget list refers to gadgets in a requester and the pointer
        must necessarily point to a window.  If these are not the gadgets
        of a requester, the requester argument may be NULL.

        Be sure to see the RefreshGadgets() function description, as this
        function is simply an extension of that.

    INPUTS
        Gadgets = pointer to the first in the list of gadgets wanting
            refreshment
        Window = pointer to the window containing the gadget or its requester
        Requester = pointer to a requester (ignored if Gadget is not attached
            to a Requester).
        NumGad  = maximum number of gadgets to be refreshed.  A value of -1
          will cause all gadgets to be refreshed from gadget to the
          end of the list.  A value of -2 will also do this, but if 'Gadgets'
          points to a Requester Gadget (GTYP_REQGADGET) ALL gadgets in the
          requester will be refreshed (this is a mode compatible with v1.1
          RefreshGadgets().)

    RESULT
        None

    BUGS

    SEE ALSO
        RefreshGadgets()
```

```
intuition.library/RefreshWindowFrame      intuition.library/RefreshWindowFrame

    NAME
        RefreshWindowFrame -- Ask Intuition to redraw your window border.

    SYNOPSIS
        RefreshWindowFrame( Window )
                            A0

        VOID RefreshWindowFrame( struct Window * );

    FUNCTION
        Refreshes the border of a window, including title region and all
        of the window's gadgets.

        You may use this call if you wish to update the display of your borders.
        The expected use of this is to correct unavoidable corruption.

    INPUTS
        Window = a pointer to a Window structure

    RESULT
        None

    BUGS

    SEE ALSO
```

```
intuition.library/ReleaseGIRPort           intuition.library/ReleaseGIRPort

    NAME
        ReleaseGIRPort -- Release a custom gadget RastPort. (V36)

    SYNOPSIS
        ReleaseGIRPort( RPort )
                        A0

        VOID ReleaseGIRPort( struct RastPort * );

    FUNCTION
        The corresponding function to ObtainGIRPort(), it releases
        arbitration used by Intuition for gadget RastPorts.

    INPUTS
        Pointer to the RastPort returned by ObtainGIRPort().
        This pointer can be NULL, in which case nothing happens.

    RESULT
        None

    BUGS

    SEE ALSO
        ObtainGIRPort(), Custom Gadget documentation
```

```
intuition.library/RemakeDisplay             intuition.library/RemakeDisplay

    NAME
        RemakeDisplay -- Remake the entire Intuition display.

    SYNOPSIS
        RemakeDisplay()

        VOID RemakeDisplay( VOID );

    FUNCTION
        This is the big one.

        This procedure remakes the entire View structure for the
        Intuition display.  It does the equivalent of MakeScreen() for
        every screen in the system, and then it calls the internal
        equivalent of RethinkDisplay().

        WARNING:  This routine can take many milliseconds to run, so
        do not use it lightly.

        Calling MakeScreen() followed by RethinkDisplay() is typically
        a more efficient method for affecting changes to a single
        screen's ViewPort.

    INPUTS
        None

    RESULT
        None

    BUGS

    SEE ALSO
        MakeScreen(), RethinkDisplay(), graphics.library/MakeVPort()
        graphics.library/MrgCop(), graphics.library/LoadView()
```

```
intuition.library/RemoveClass                    intuition.library/RemoveClass

   NAME
       RemoveClass -- Make a public boopsi class unavailable. (V36)

   SYNOPSIS
       RemoveClass( classPtr )
                    A0

       VOID RemoveClass( struct IClass * );

   FUNCTION
       Makes a public class unavailable for public consumption.
       It's OK to call this function for a class which is not
       yet in the internal public class list, or has been
       already removed.

   INPUTS
       ClassPtr = pointer to *public* class created by MakeClass(),
               may be NULL.

   RESULT
       None.

   NOTES

   BUGS

   SEE ALSO
       MakeClass(), FreeClass(), AddClass()
       Document "Basic Object-Oriented Programming System for Intuition"
       and the "boopsi Class Reference" document.
```

```
intuition.library/RemoveGadget                  intuition.library/RemoveGadget

   NAME
       RemoveGadget -- Remove a gadget from a window.

   SYNOPSIS
       Position = RemoveGadget( Window, Gadget )
       D0                       A0      A1

       UWORD RemoveGadget( struct Window *, struct Gadget * );

   FUNCTION
       Removes the given gadget from the gadget list of the specified
       window.  Returns the ordinal position of the removed gadget.

       If the gadget is in a requester attached the the window, this
       routine will look for it and remove it if it is found.

       If the gadget pointer points to a gadget that isn't in the
       appropriate list, -1 is returned.  If there aren't any gadgets in the
       list, -1 is returned.  If you remove the 65535th gadget from the list
       -1 is returned.

   NOTES
       New with V37: If one of the gadgets you wish to remove
       is the active gadget, this routine will wait for the user
       to release the mouse button before deactivating and removing
       the gadget.

   INPUTS
       Window = pointer to the window containing the gadget or the requester
           containing the gadget to be removed.
       Gadget = pointer to the gadget to be removed.  The gadget itself
           describes whether this is a gadget that should be removed from the
           window or some requester.

   RESULT
       Returns the ordinal position of the removed gadget.  If the gadget
       wasn't found in the appropriate list, or if there are no gadgets in
       the list, returns -1.

   BUGS

   SEE ALSO
       AddGadget(), AddGList(), RemoveGList()
```

```
intuition.library/RemoveGList                     intuition.library/RemoveGList

    NAME
        RemoveGList -- Remove a sublist of gadgets from a window.

    SYNOPSIS
        Position = RemoveGList( Window, Gadget, Numgad )
        D0                      A0      A1      D0

        UWORD RemoveGList( struct Window *, struct Gadget *, WORD );

    FUNCTION
        Removes 'Numgad' gadgets from the gadget list of the specified
        window.  Will remove gadgets from a requester if the first
        gadget's GadgetType flag GTYP_REQGADGET is set.

        Otherwise identical to RemoveGadget().

    NOTE
        The last gadget in the list does NOT have its link zeroed.
        New with V36: OK, last gadget's NextGadget field is set to NULL.

        New with V37: If one of the gadgets you wish to remove
        is the active gadget, this routine will wait for the user
        to release the mouse button before deactivating and removing
        the gadget.

    INPUTS
        Window = pointer to the window containing the gadget or the requester
            containing the gadget to be removed.
        Gadget = pointer to the gadget to be removed.  The gadget itself
           describes whether this is a gadget that should be removed
           from the window or some requester.
        Numgad = number of gadgets to be removed.  If -1, remove all gadgets
            to end of window gadget list

    RESULT
        Returns the ordinal position of the removed gadget.  If the gadget
        wasn't found in the appropriate list, or if there are no gadgets in
        the list, returns -1.

    BUGS

    SEE ALSO
        RemoveGadget(), AddGadget(), AddGList()
```

```
intuition.library/ReportMouse                     intuition.library/ReportMouse
    NAME
        ReportMouse -- Tell Intuition whether to report mouse movement.

    SYNOPSIS
        ReportMouse( Boolean, Window )
                     D0       A0        <-note

        VOID ReportMouse( BOOL, struct Window * );

    SPECIAL NOTE
        Some compilers and link files switch the arguments to this function
        about in unpredictable ways.  We apologize for this confusion
        wrapped around an error enclosing a mistake.
        The call will take one of two forms:

                ReportMouse(Boolean, Window);
                        -or-
                ReportMouse(Window, (ULONG)Boolean);

        The first form is the one that corresponds to the amiga.lib supplied
        by Commodore.  The linker libraries and "pragmas" of some compilers
        supply the alternate form.

        A key point to remember is that the form of the function in ROM
        has always been the same, so there has always been object code
        compatibility.  However some care should be taken when switching
        compilers or switching between stubs and pragmas.

        From assembler the interface has always been:
                Boolean in D0, Window in A0

        Also, it is still endorsed to simply set the WFLG_REPORTMOUSE flag bit
        in Window->Flags, or reset it, on your own.  Make the operation
        an atomic assembly instruction (OR.W #WFLG_REPORTMOUSE,wd_Flags+2(A0)
        where A0 contains your window pointer).  Most compilers will produce
        an atomic operation when faced with:
                        Window->Flags |= WFLG_REPORTMOUSE;
                        Window->Flags &=~WFLG_REPORTMOUSE;
        or else bracket the operation between Forbid()/Permit().

    FUNCTION
        Tells Intuition whether or not to broadcast mouse-movement events to
        your window when it's the active one.  The Boolean value specifies
        whether to start or stop broadcasting position information of
        mouse-movement.  If the window is the active one, mouse-movement reports
        start coming immediately afterwards.  This same routine will change
        the current state of the GACT_FOLLOWMOUSE function of a
        currently-selected gadget too.

        Note that calling ReportMouse() when a gadget is selected will only
        temporarily change whether or not mouse movements are reported while
        that gadget remains selected; the next time the gadget is selected, its
        GACT_FOLLOWMOUSE flag is examined anew.

        Note also that calling ReportMouse() when no gadget is currently
        selected will change the state of the window's WFLG_REPORTMOUSE flag,
        but will have no effect on any gadget that may be subsequently
        selected. (This is all fixed in V36.)

        The ReportMouse() function is first performed when OpenWindow()
        is first called; if the flag WFLG_REPORTMOUSE is included among
        the options, then all mouse-movement events are reported
        to the opening task and will continue to be reported
        until ReportMouse() is called with a Boolean value of FALSE.
        If WFLG_REPORTMOUSE is not set, then no mouse-movement reports will
    be broadcast until ReportMouse() is called with a Boolean of TRUE.

    Note that the WFLG_REPORTMOUSE flag, as managed by this routine,
    determines IF mouse messages are to be broadcast.  Determining HOW
    they are to be broadcast is determined by the IDCMP_MOUSEMOVE
    IDCMPFlag.

INPUTS
    Window = pointer to a Window structure associated with this request
    Boolean = TRUE or FALSE value specifying whether to turn this
         function on or off

RESULT
    None

BUGS
    See above

SEE ALSO
    The Input and Output section of the Intuition Reference Manual
```

```
intuition.library/Request                          intuition.library/Request

   NAME
       Request -- Activate a requester.

   SYNOPSIS
       Success = Request( Requester, Window )
       D0                 A0         A1

       BOOL Request( struct Requester *, struct Window * );

   FUNCTION
       Links in and displays a requester into the specified window.

       This routine ignores the window's IDCMP_REQVERIFY flag.

   INPUTS
       Requester = pointer to the requester to be displayed
       Window = pointer to the window into which this requester goes

       New for V36: the POINTREL flag now has meaning if the requester
       is not a DMR (Double-Menu Requester):
       If POINTREL is set this requester should be placed in the center
       of the window, offset by Requester.RelLeft and Requester.RelTop.
       If the requester doesn't fit in the window, its position will be
       adjusted to show the upper-left corner.

   RESULT
       If the requester is successfully opened, TRUE is returned.  Otherwise,
       if the requester could not be opened, FALSE is returned.

   BUGS
       There is a maximum of 8 requesters that are supported in
       a window that can be changed in size, position, or depth.

   SEE ALSO
       The Requesters section of the Intuition Reference Manual
```

intuition.library

```
intuition.library/ResetMenuStrip             intuition.library/ResetMenuStrip

    NAME
        ResetMenuStrip -- Re-attach a menu strip to a window. (V36)

    SYNOPSIS
        Success = ResetMenuStrip( Window, Menu )
        D0                        A0      A1

        BOOL ResetMenuStrip( struct Window *, struct Menu * );

    FUNCTION
        This function is simply a "fast" version of SetMenuStrip() that
        doesn't perform the precalculations of menu page sizes that
        SetMenuStrip() does.

        You may call this function ONLY IF the menu strip and all items
        and sub-items have not changed since the menu strip was passed to
        SetMenuStrip(), with the following exceptions:

        - You may change the CHECKED flag to turn a checkmark on or off.
        - You may change the ITEMENABLED flag to enable/disable some
          MenuItem or Menu structures.

        In all other ways, this function performs like SetMenuStrip().

        The new sequence of events you can use is:
        - OpenWindow()
        - SetMenuStrip()
        zero or more iterations of:
            - ClearMenuStrip()
            - change CHECKED or ITEMENABLED flags
            - ResetMenuStrip()
        - ClearMenuStrip()
        - CloseWindow()

    INPUTS
        Window = pointer to a Window structure
        Menu = pointer to the first menu in the menu strip

    RESULT
        TRUE always.

    BUGS

    SEE ALSO
        SetMenuStrip(), ClearMenuStrip()
```

intuition.library

```
intuition.library/RethinkDisplay             intuition.library/RethinkDisplay

    NAME
        RethinkDisplay -- Grand manipulation of the entire Intuition display.

    SYNOPSIS
        RethinkDisplay()

        VOID RethinkDisplay( VOID );

    FUNCTION
        This function performs the Intuition global display reconstruction.
        This includes rethinking about all of the  ViewPorts and their
        relationship to one another and reconstructing the entire display
        based on the results of this rethinking.

        Specifically, and omitting many internal details, the operation
        consists of this:

            Determine which ViewPorts are invisible and set their VP_HIDE
            ViewPort Mode flag. VP_HIDE flags are also set for screens that
            may not be simultaneously displayed with the frontmost (V36).

            If a change to a viewport height, or changing interlace or
            monitor scan rates require, MakeVPort() is called for specific
            screen viewports.  After this phase, the intermediate Copper lists
            for each screen's viewport are correctly set up.

            MrgCop() and LoadView() are then called to get these Copper lists
            in action, thus establishing the new state of the Intuition
            display.

        You may perform a MakeScreen() on your Custom Screen before calling
        this routine.  The results will be incorporated in the new display, but
        changing the INTERLACE ViewPort mode for one screens must be reflected
        in the Intuition View, which is left to Intuition.

        WARNING:  This routine can take several milliseconds to run, so
        do not use it lightly.

        New for V36: This routine is substantially changed to support
        new screen modes.  In particular, if screen rearrangement has
        caused a change in interlace mode or scan rate, this routine
        will remake the copper lists for each screen's viewport.

    INPUTS
        None

    RESULT
        None

    BUGS
        In V35 and earlier, an interlaced screen coming to the front
        may not trigger a complete remake as required when the global
        interlace state is changed.  In some cases, this can be compensated
        for by setting the viewport DHeight field to 0 for hidden screens.

    SEE ALSO
        RemakeDisplay(), graphics.library/MakeVPort(), graphics.library/MrgCop(),
        graphics.library/LoadView(), MakeScreen()
```

intuition.library Page 117

```
intuition.library/ScreenToBack                    intuition.library/ScreenToBack

    NAME
        ScreenToBack -- Send the specified screen to the back of the display.

    SYNOPSIS
        ScreenToBack( Screen )
                      A0

        VOID ScreenToBack( struct Screen * );

    FUNCTION
        Sends the specified screen to the back of the display.

    INPUTS
        Screen = pointer to a Screen structure

    RESULT
        None

    BUGS

    SEE ALSO
        ScreenToFront()
```

intuition.library Page 118

```
intuition.library/ScreenToFront                  intuition.library/ScreenToFront

    NAME
        ScreenToFront -- Make the specified screen the frontmost.

    SYNOPSIS
        ScreenToFront( Screen )
                       A0

        VOID ScreenToFront( struct Screen * );

    FUNCTION
        Brings the specified Screen to the front of the display.

    INPUTS
        Screen = a pointer to a Screen structure

    RESULT
        None

    BUGS

    SEE ALSO
        ScreenToBack()
```

```
intuition.library/SetAttrsA                             intuition.library/SetAttrsA

    NAME
        SetAttrsA -- Specify attribute values for an object. (V36)
        SetAttrs -- Varargs stub for SetAttrsA(). (V36)

    SYNOPSIS
        result = SetAttrsA( Object, TagList )
        D0                  A0      A1

        ULONG SetAttrsA( APTR, struct TagItem * );

        result = SetAttrs( Object, Tag1, ... )

        ULONG SetAttrs( APTR, ULONG, ... );

    FUNCTION
        Specifies a set of attribute/value pairs with meaning as
        defined by a 'boopsi' object's class.

        This function does not provide enough context information or
        arbitration for boopsi gadgets which are attached to windows
        or requesters.  For those objects, use SetGadgetAttrs().

    INPUTS
        Object = abstract pointer to a boopsi object.
        TagList = array of TagItem structures with attribute/value pairs.

    RESULT
        The object does whatever it wants with the attributes you provide.
        The return value tends to be non-zero if the changes would require
        refreshing gadget imagery, if the object is a gadget.

    NOTES
        This function invokes the OM_SET method with a NULL GadgetInfo
        parameter.

    BUGS

    SEE ALSO
        NewObject(), DisposeObject(), GetAttr(), MakeClass(),
        Document "Basic Object-Oriented Programming System for Intuition"
        and the "boopsi Class Reference" document.
```

```
intuition.library/SetDefaultPubScreen   intuition.library/SetDefaultPubScreen

    NAME
        SetDefaultPubScreen -- Choose a new default public screen. (V36)

    SYNOPSIS
        SetDefaultPubScreen( Name )
                             A0

        VOID SetDefaultPubScreen( UBYTE * );

    FUNCTION
        Establishes a new default public screen for visitor windows.

        This screen is used by windows asking for a named public screen
        that doesn't exist and the FALLBACK option is selected, and for
        windows asking for the default public screen directly.

    INPUTS
        Name = name of chosen public screen to be the new default.
        A value of NULL means that the Workbench screen is to
        be the default public screen.

    RESULT
        None

    BUGS

    SEE ALSO
        OpenWindow(), OpenScreen(), Intuition V36 update documentation
```

```
intuition.library/SetDMRequest                   intuition.library/SetDMRequest

    NAME
        SetDMRequest -- Set the DMRequest of a window.

    SYNOPSIS
        success = SetDMRequest( Window, DMRequest )
        D0                      A0      A1

        BOOL SetDMRequest( struct Window *, struct Requester * );

    FUNCTION
        Attempts to set the DMRequest into the specified window.
        The DMRequest is the special requester that you attach to
        the double-click of the menu button which the user can then
        bring up on demand.  This routine WILL NOT change the DMRequest
        if it's already set and is currently active (in use by the user).
        After having called SetDMRequest(), if you want to change the
        DMRequest, the correct way to start is by calling ClearDMRequest()
        until it returns a value of TRUE; then you can call SetDMRequest()
        with the new DMRequest.

        If the POINTREL flag is set in the DMRequest, the DMR will open as
        close to the pointer as possible.  The RelLeft/Top fields are
        for fine-tuning the position.

    INPUTS
        Window = pointer to the window from which the DMRequest is to be set
        DMRequest = a pointer to a requester

    RESULT
        If the current DMRequest was not in use, sets the DMRequest
            pointer into the window and returns TRUE.
        If the DMRequest was currently in use, doesn't change the pointer
            and returns FALSE

    BUGS

    SEE ALSO
        ClearDMRequest(), Request()
```

```
intuition.library/SetEditHook                     intuition.library/SetEditHook

    NAME
        SetEditHook -- Set global processing for string gadgets. (V36)

    SYNOPSIS
        OldHook = SetEditHook( Hook )
        D0                     A0

        struct Hook *SetEditHook( struct Hook * );

    FUNCTION
        Sets new global editing hook for string gadgets.

        WARNING: This function is wholly untested.  Do *NOT* use
        this in a commercial product until further notice.

    INPUTS
        Hook -- A pointer to a struct Hook which determines a function
          in your code to be called every time the user types a key.
          This is done before control is passed to the gadget custom
          editing hook, so effects ALL string gadgets.

    RESULT
        Returns previous global edit hook structure.

    BUGS
        Unknown, risky.

    SEE ALSO
```

intuition.library Page 123

```
intuition.library/SetGadgetAttrsA           intuition.library/SetGadgetAttrsA

    NAME
        SetGadgetAttrsA -- Specify attribute values for a boopsi gadget. (V36)
        SetGadgetAttrs -- Varargs stub for SetGadgetAttrsA(). (V36)

    SYNOPSIS
        result = SetGadgetAttrsA( Gadget, Window, Requester, TagList )
        D0                        A0      A1      A2         A3

        ULONG SetGadgetAttrsA( struct Gadget *, struct Window *,
                struct Requester *, struct TagItem * );

        result = SetGadgetAttrs( Gadget, Window, Requester, Tag1, ...)

        ULONG SetGadgetAttrs( struct Gadget *, struct Window *,
                struct Requester *, ULONG, ... );

    FUNCTION
        Same as SetAttrs(), but provides context information and
        arbitration for classes which implement custom Intuition gadgets.

        You should use this function for boopsi gadget objects which have
        already been added to a requester or a window, or for "models" which
        propagate information to gadget already added.

        Typically, the gadgets will refresh their visuals to reflect
        changes to visible attributes, such as the value of a slider,
        the text in a string-type gadget, the selected state of a button.

        You can use this as a replacement for SetAttrs(), too, if you
        specify NULL for the 'Window' and 'Requester' parameters.

    INPUTS
        Gadget = abstract pointer to a boopsi gadget
        Window = window gadget has been added to using AddGList() or AddGadget()
        Requester = for REQGADGETs, requester containing the gadget
        TagList = array of TagItem structures with attribute/value pairs.

    RESULT
        The object does whatever it wants with the attributes you provide,
        which might include updating its gadget visuals.

        The return value tends to be non-zero if the changes would require
        refreshing gadget imagery, if the object is a gadget.

    NOTES
        This function invokes the OM_SET method with a GadgetInfo
        derived from the 'Window' and 'Requester' pointers.

    BUGS
        There should be more arbitration between this function and
        the calls that Intuition's input task will make to the
        gadgets.  In the meantime, this function, input processing,
        and refreshing must be mutually re-entrant.

    SEE ALSO
        NewObject(), DisposeObject(), GetAttr(), MakeClass(),
        Document "Basic Object-Oriented Programming System for Intuition"
        and the "boopsi Class Reference" document.
```

intuition.library Page 124

```
intuition.library/SetMenuStrip                 intuition.library/SetMenuStrip

    NAME
        SetMenuStrip -- Attach a menu strip to a window.

    SYNOPSIS
        Success = SetMenuStrip( Window, Menu )
        D0                      A0      A1

        BOOL SetMenuStrip( struct Window *, struct Menu * );

    FUNCTION
        Attaches the menu strip to the window.  After calling this routine,
        if the user presses the menu button, this specified menu strip
        will be displayed and accessible by the user.

        Menus with zero menu items are not allowed.

        NOTE:  You should always design your menu strip changes to be a
        two-way operation, where for every menu strip you add to your
        window you should always plan to clear that strip sometime.  Even
        in the simplest case, where you will have just one menu strip for
        the lifetime of your window, you should always clear the menu strip
        before closing the window.  If you already have a menu strip attached
        to this window, the correct procedure for changing to a new menu
        strip involves calling ClearMenuStrip() to clear the old first.

        The sequence of events should be:
        - OpenWindow()
        - zero or more iterations of:
        - SetMenuStrip()
        - ClearMenuStrip()
        - CloseWindow()

    INPUTS
        Window = pointer to a Window structure
        Menu = pointer to the first menu in the menu strip

    RESULT
        TRUE if there were no problems.  TRUE always, since this routine
        will wait until it is OK to proceed.

    BUGS

    SEE ALSO
        ClearMenuStrip(), ResetMenuStrip()
```

```
intuition.library/SetMouseQueue             intuition.library/SetMouseQueue

    NAME
        SetMouseQueue -- Change limit on pending mouse messages. (V36)

    SYNOPSIS
        oldQueueLength = SetMouseQueue( Window, QueueLength )
        D0                              A0      D0

        LONG SetMouseQueue( struct Window *, UWORD );

    FUNCTION
        Changes the number of mouse messages that Intuition will allow
        to be outstanding for your window.

    INPUTS
        Window = your window
        QueueLength = the new value of outstanding mouse movement messages
          you wish to allow.

    RESULT
        -1 if 'Window' is not known
        Otherwise the previous value of the queue limit.
        The corresponding function for changing the repeat key
        queue limit is not yet implemented.

    BUGS

    SEE ALSO
        OpenWindow()
```

```
intuition.library/SetPointer                   intuition.library/SetPointer

    NAME
        SetPointer -- Specify a pointer sprite image for a window.

    SYNOPSIS
        SetPointer( Window, Pointer, Height, Width, XOffset, YOffset )
                    A0      A1       D0      D1     D2       D3

        VOID SetPointer( struct Window *, UWORD *, WORD, WORD, WORD, WORD );

    FUNCTION
        Sets up the window with the sprite definition for the pointer.
        Then, whenever the window is the active one, the pointer
        image will change to the window's version.  If the window is
        the active one when this routine is called, the change takes
        place immediately.

        The XOffset and YOffset parameters are used to offset the
        upper-left corner of the hardware sprite image from what Intuition
        regards as the current position of the pointer.  Another way of
        describing it is as the offset from the "hot spot" of the pointer
        to the top-left corner of the sprite.  For instance, if you specify
        offsets of zero, zero, then the top-left corner of your sprite
        image will be placed at the mouse position.  On the other hand,
        if you specify an XOffset of -7 (remember, sprites are 16 pixels
        wide) then your sprite will be centered over the mouse position.
        If you specify an XOffset of -15, the right-edge of the sprite
        will be over the mouse position.

    INPUTS
        Window = pointer to the window to receive this pointer definition
        Pointer = pointer to the data definition of a sprite
        Height = the height of the pointer
        Width = the width of the sprite (must be less than or equal to sixteen)
        XOffset = the offset for your sprite from the mouse position
        YOffset = the offset for your sprite from the mouse position

    RESULT
        None

    BUGS

    SEE ALSO
        ClearPointer(), Intuition Reference Manual or Amiga Rom
        Kernel Manual
```

```
intuition.library/SetPrefs                              intuition.library/SetPrefs

   NAME
       SetPrefs -- Set Intuition preferences data.

   SYNOPSIS
       Prefs = SetPrefs( PrefBuffer, Size, Inform )
       D0                A0          D0    D1

       struct Preferences *SetPrefs( struct Preferences *, LONG, BOOL );

   FUNCTION
       Sets new preferences values.  Copies the first 'Size' bytes
       from your preferences buffer to the system preferences table,
       and puts them into effect.

       The 'Inform' parameter, if TRUE, indicates that an IDCMP_NEWPREFS
       message is to be sent to all windows that have the IDCMP_NEWPREFS
       IDCMPFlag set.

       It is legal to set a partial copy of the Preferences structure.
       The most frequently changed values are grouped at the beginning
       of the Preferences structure.

       New for V36:  A new and more extensible method for supplying
       Preferences has been introduced in V36, and relies on file
       system notification.  The Intuition preferences items rely
       also on the IPrefs program.  Certain elements of the
       Preferences structure have been superceded by this new method.
       Pointer changes submitted through SetPrefs() are only heeded
       until the first time IPrefs informs Intuition of a V36-style
       pointer.ilbm preferences file.  The Preferences FontHeight and
       LaceWB fields are respected only from the system-configuration
       file, and never thereafter.  As well, the view centering and
       size apply only to the default monitor, and not to such modes
       as Productivity.  Other fields may be superceded in the future.

   INPUTS
       PrefBuffer = pointer to the memory buffer which contains your
           desired settings for Intuition preferences
       Size = the number of bytes in your PrefBuffer, the number of bytes
          you want copied to the system's internal preference settings
       Inform = whether you want the information of a new preferences
           setting propagated to all windows.

   NOTES
       Unless you are responding to a user's explicit request to
       change Preferences (for example, you are writing a Preferences
       editor), you should probably avoid using this function.
       The user's Preferences should be respected, not overridden.

   RESULT
       Returns your parameter PrefBuffer.

   BUGS

   SEE ALSO
       GetDefPrefs(), GetPrefs()
```

```
intuition.library/SetPubScreenModes        intuition.library/SetPubScreenModes

   NAME
       SetPubScreenModes -- Establish global public screen behavior. (V36)

   SYNOPSIS
       OldModes = SetPubScreenModes( Modes )
       D0                            D0

       UWORD SetPubScreenModes( UWORD );

   FUNCTION
       Sets GLOBAL Intuition public screen modes.

   INPUTS
       Modes = new global modes flags.  Values for flag bits are:
         SHANGHAI: workbench windows are to be opened on the
               default public screen
         POPPUBSCREEN: when a visitor window is opened, the public
               screen it opens on is to be brought to the front.

   RESULT
       OldModes = previous global mode settings

   BUGS

   SEE ALSO
       OpenScreen(), Intuition V36 update documentation
```

```
intuition.library/SetWindowTitles          intuition.library/SetWindowTitles

   NAME
       SetWindowTitles -- Set the window's titles for both window and screen.

   SYNOPSIS
       SetWindowTitles( Window, WindowTitle, ScreenTitle )
                        A0      A1           A2

       VOID SetWindowTitles( struct Window *, UBYTE *, UBYTE * );

   FUNCTION
       Allows you to set the text which appears in the Window and/or Screen
       title bars.

       The window title appears at all times along the window title bar.
       The window's screen title appears at the screen title bar whenever
       this window is the active one.

       When this routine is called, your window title will be changed
       immediately.  If your window is the active one when this routine is
       called, the screen title will be changed immediately.

       You can specify a value of -1 (i.e. (UBYTE *) ~0) for either of
       the title pointers.  This designates that you want Intuition to leave
       the current setting of that particular title alone, and modify
       only the other one.  Of course, you could set both to -1.

       Furthermore, you can set a value of 0 (zero) for either of the
       title pointers.  Doing so specifies that you want no title to
       appear (the title bar will be blank).

       Both of the titles are rendered in the default font of the window's
       screen, as set using OpenScreen().

       In setting the window's title, Intuition may do some other rendering
       in the top border of your window.  If your own rendering sometimes
       appears in your window border areas, you may want to restore the entire
       window border frame.  The function SetWindowTitles() does not do this
       in the newer versions.  The function RefreshWindowFrame() is provided
       to do this kind of thing for you.

   INPUTS
       Window = pointer to your window structure
       WindowTitle = pointer to a null-terminated text string, or set to
          either the value of -1 (negative one) or 0 (zero)
       ScreenTitle = pointer to a null-terminated text string, or set to
          either the value of -1 (negative one) or 0 (zero)

   RESULT
       None

   BUGS

   SEE ALSO
       OpenWindow(), RefreshWindowFrame(), OpenScreen()
```

```
intuition.library/ShowTitle                          intuition.library/ShowTitle

   NAME
       ShowTitle -- Set the screen title bar display mode.

   SYNOPSIS
       ShowTitle( Screen, ShowIt )
                  A0      D0

       VOID ShowTitle( struct Screen *, BOOL );

   FUNCTION
       This routine sets the SHOWTITLE flag of the specified screen, and
       then coordinates the redisplay of the screen and its windows.

       The screen title bar can appear either in front of or behind
       WFLG_BACKDROP windows.  This is contrasted with the fact that
       non-WFLG_BACKDROP windows always appear in front of the screen title
       bar.  You specify whether you want the screen title bar to be in front
       of or behind the screen's WFLG_BACKDROP windows by calling this
       routine.

       The ShowIt argument should be set to either TRUE or FALSE.  If TRUE,
      the screen's title bar will be shown in front of WFLG_BACKDROP windows

      If FALSE, the title bar will be rendered behind all windows.

      When a screen is first opened, the default setting of the SHOWTITLE
      flag is TRUE.

   INPUTS
      Screen = pointer to a Screen structure
      ShowIt = Boolean TRUE or FALSE describing whether to show or hide the
          screen title bar

   RESULT
      None

   BUGS

   SEE ALSO
```

```
intuition.library/SizeWindow                    intuition.library/SizeWindow

    NAME
        SizeWindow -- Ask Intuition to size a window.

    SYNOPSIS
        SizeWindow( Window, DeltaX, DeltaY )
                    A0      D0      D1

        VOID SizeWindow( struct Window *, WORD, WORD );

    FUNCTION
        This routine sends a request to Intuition asking to size the window
        the specified amounts.  The delta arguments describe how much to
        size the window along the respective axes.

        Note that the window will not be sized immediately, but rather
        will be sized the next time Intuition receives an input event,
        which happens currently at a minimum rate of ten times per second,
        and a maximum of sixty times a second.  You can discover when
        you window has finally been sized by setting the IDCMP_NEWSIZE flag
        of the IDCMP of your window.  See the "Input and Output Methods"
        chapter of The Intuition Reference Manual for description of the IDCMP.

        New for V36: Intuition now will do validity checking on the final
        dimensions.  To change to new absolute dimensions, or to move and
        size a window in one step, use ChangeWindowBox().

        However, limit checking against window MinWidth, MinHeight,
        MaxWidth, and MaxHeight was not done prior to V36, and
        these fields are still ignored (as documented) if you have
        no sizing gadget (WFLG_SIZEGADGET is not set).  The *are*
        respected now (V36) if WFLG_SIZEGADGET is set.

        New for V36: you can determine when the change in size has
        taken effect by receiving the IDCMP_CHANGEWINDOW IDCMP message.

    INPUTS
        Window = pointer to the structure of the window to be sized
        DeltaX = signed value describing how much to size the window
         on the x-axis
        DeltaY = signed value describing how much to size the window
         on the y-axis

    RESULT
        None

    BUGS

    SEE ALSO
        ChangeWindowBox(), MoveWindow(), WindowToFront(), WindowToBack()
```

```
intuition.library/SysReqHandler              intuition.library/SysReqHandler

    NAME
        SysReqHandler -- Handle system requester input. (V36)

    SYNOPSIS
        num = SysReqHandler( Window, IDCMPFlagsPtr, WaitInput )
        D0                   A0      A1             D0

        LONG SysReqHandler( struct Window *, ULONG *, BOOL );

    FUNCTION
        Handles input for a window returned by either BuildSysRequest()
        or BuildEasyRequest().  These functions with SysReqHandler()
        you can perform an "asynchronous" EasyRequest() or AutoRequest().
        That is to say, you can perform other processing while you
        wait for the requester to be satisfied.

        Each time this function is called, it will process all
        IDCMPMessages that the window has received.  If the parameter
        'WaitInput' is non-zero, SysReqHandler() will wait for input
        (by calling WaitPort()) if there are no IDCMP messages.

        SysReqHandler() returns the same values as EasyRequest(): A gadget
        ID greater than equal to 0, and  -1 if one of the other IDCMP
        events were received.

        An additional value of -2 is returned if the input processed
        does not satisfy the requester.  In this case, you might
        perform some processing and call SysReqHandler() again.

        Note: this function does NOT terminate the system request.
        Not only must you call FreeSysRequest() to eliminate the request,
        but you may also continue processing after an event which would
        normally terminate a normal call to EasyRequest().

    EXAMPLE
        Implementation of EasyRequest() input loop:

        window = BuildEasyRequest( ZZZZ )
        while ( (retval = SysReqHandler( window, idcmp_ptr, TRUE )) == -2 )
        {
            /* loop     */;
        }
        FreeSysRequest( window );

    EXAMPLE
        Request a volume, but don't remove the requester when the
        user inserts the wrong disk:

        struct EasyStruct volumeES = {
            sizeof (struct EasyStruct),
            0,
            "Volume Request",
            "Please insert volume %s in any drive.",
            "Cancel"
        };

        Volume *
        getVolume( volname )
        UBYTE   *volname;
        {
            struct Window       *window;
            Volume              *volume = NULL;
            Volume              *findVolume();
            int                 retval;
        window = BuildEasyRequest( NULL, &volumeES, IDCMP_DISKINSERTED,
                volname );

        while ( (retval = SysReqHandler( window, NULL, TRUE )) != 0 )
        {
            /* not cancelled yet    */

            /* when IDCMP_DISKINSERTED, check for volume */
            if (( retval == -1 ) && (volume = findVolume( volname )))
                break;
        }
        FreeSysRequest( window );
        return ( volume );
    }

INPUTS
    Window = Window pointer returned from BuildSysRequest() or
            BuildEasyRequest().  Those functions can also return
            values '0' or '1', and these values may also be
            passed to SysReqHandler(), which will immediately
            return the same value.

    IDCMPFlagsPtr = If you passed application specific IDCMP
            flags to BuildSysRequest() or BuildEasyRequest(),
            SysReqHandler() will return -1 if that IDCMP message
            is received.  If IDCMPFlagsPtr is non-null, it
            points to a ULONG where the IDCMP class received
            will be copied for your examination.

            This pointer can be NULL if you have provided no
            application specific IDCMP flags or if you do
            not need to know which application specific IDCMP
            event occurred.

            If you provide more than on flag in the flags variable
            this pointer points to, you will have to refresh
            the variable whenever -1 is returned, since the
            variable will have been changed to show just the
            single IDCMP Class bit that caused the return.

    WaitInput = Specifies that you want SysReqHandler() to
            to wait for IDCMP input if there is none pending.

RESULT
    0, 1, ..., N = Successive GadgetID values, for the gadgets
        you specify for the requester.  NOTE: The numbering
        from left to right is actually: 1, 2, ..., N, 0.
        This is for compatibility with AutoRequests which has
        FALSE for the rightmost gadget.

    -1 = Means that one of the caller-supplied IDCMPFlags occurred.
        The IDCMPFlag value is in the longword pointed to by UDCMP_ptr.

    -2 = input processed did not satisfy the requester. One example
        is a keystroke that does not satisfy the requester.  Another
        example is if there is no input pending and you specified
        FALSE for WaitInput.

BUGS

SEE ALSO
    exec.library/WaitPort()
```

```
intuition.library/UnlockIBase                   intuition.library/UnlockIBase

    NAME
        UnlockIBase -- Surrender an Intuition lock gotten by LockIBase().

    SYNOPSIS
        UnlockIBase( Lock )
                     A0

        VOID UnlockIBase( ULONG );

    FUNCTION

        Surrenders lock gotten by LockIBase().

        Calling this function when you do not own the specified lock will
        immediately crash the system.

    INPUTS
        The value returned by LockIBase() should be passed to this function,
        to specify which internal lock is to be freed.

        Note that the parameter is passed in A0, not D0, for historical reasons.

    RESULT
        None

    BUGS

    SEE ALSO
        LockIBase()
```

```
intuition.library/UnlockPubScreen          intuition.library/UnlockPubScreen

    NAME
        UnlockPubScreen -- Release lock on a public screen. (V36)

    SYNOPSIS
        UnlockPubScreen( Name, [Screen] )
                         A0     A1

        VOID UnlockPubScreen( UBYTE *, struct Screen * );

    FUNCTION
        Releases lock gotten by LockPubScreen().
        It is best to identify the locked public screen by
        the pointer returned from LockPubScreen().  To do this,
        supply a NULL 'Name' pointer and the screen pointer.

        In rare circumstances where it would be more convenient to pass
        a non-NULL pointer to the public screen name string, the
        'Screen' parameter is ignored.

    INPUTS
        Name = pointer to name of public screen.  If Name is NULL,
            then argument 'Screen' is used as a direct pointer to
            a public screen.
        Screen = pointer to a public screen.  Used only if Name
            is NULL.  This pointer MUST have been returned
            by LockPubScreen().

    RESULT

    BUGS

    SEE ALSO
        LockPubScreen()
```

```
intuition.library/UnlockPubScreenList    intuition.library/UnlockPubScreenList

    NAME
        UnlockPubScreenList -- Release public screen list semaphore. (V36)

    SYNOPSIS
        UnlockPubScreenList()

        VOID UnlockPubScreenList( VOID );

    FUNCTION
        Releases lock gotten by LockPubScreenList().

    INPUTS
        None.

    RESULT
        None.

    BUGS

    SEE ALSO
        LockPubScreenList()
```

```
intuition.library/ViewAddress                    intuition.library/ViewAddress

    NAME
        ViewAddress -- Return the address of the Intuition View structure.

    SYNOPSIS
        view = ViewAddress()
        D0

        struct View *ViewAddress( VOID );

    FUNCTION
        Returns the address of the Intuition View structure.  If you
        want to use any of the graphics, text, or animation primitives
        in your window and that primitive requires a pointer to a view,
        this routine will return the address of the view for you.

    INPUTS
        None

    RESULT
        Returns the address of the Intuition View structure

    BUGS

    SEE ALSO
        graphics.library
```

```
intuition.library/ViewPortAddress            intuition.library/ViewPortAddress

    NAME
        ViewPortAddress -- Return the address of a window's viewport.

    SYNOPSIS
        ViewPort = ViewPortAddress( Window )
        D0                          A0

        struct ViewPort *ViewPortAddress( struct Window * );

    FUNCTION
        Returns the address of the viewport associated with the specified
        window.  The viewport is actually the viewport of the screen within
        which the window is displayed.  If you want to use any of the graphics,
        text, or animation primitives in your window and that primitive
        requires a pointer to a viewport, you can use this call.

        This pointer is only valid as long as your window's screen remains
        open, which is ensured by keeping your window open.

    INPUTS
        Window = pointer to the window for which you want the viewport address

    RESULT
        Returns the address of the Intuition ViewPort structure for
        your window's screen .

    BUGS
        This routine is unnecessary: you can just use the expression
        &Window->WScreen->ViewPort.

    SEE ALSO
        graphics.library
```

intuition.library Page 139

```
intuition.library/WBenchToBack                intuition.library/WBenchToBack

    NAME
        WBenchToBack -- Send the Workbench screen in back of all screens.

    SYNOPSIS
        Success = WBenchToBack()
        D0

        BOOL WBenchToBack( VOID );

    FUNCTION
        Causes the Workbench screen, if it's currently opened, to go behind
        all other screens.  This does not 'move' the screen up or down,
        instead only affects the depth-arrangement of the screens.

    INPUTS
        None

    RESULT
        If the Workbench screen was opened, this function returns TRUE,
        otherwise it returns FALSE.

    BUGS

    SEE ALSO
        WBenchToFront(), ScreenToFront()
```

intuition.library Page 140

```
intuition.library/WBenchToFront              intuition.library/WBenchToFront

    NAME
        WBenchToFront -- Bring the Workbench screen in front of all screens.

    SYNOPSIS
        Success = WBenchToFront()
        D0

        BOOL WBenchToFront( VOID );

    FUNCTION
        Causes the Workbench Screen, if it's currently opened, to come to
        the foreground.  This does not 'move' the screen up or down, instead
        only affects the depth-arrangement of the screen.

    INPUTS
        None

    RESULT
        If the Workbench screen was opened, this function returns TRUE,
        otherwise it returns FALSE.

    BUGS

    SEE ALSO
        WBenchToBack(), ScreenToBack()
```

```
intuition.library/WindowLimits                    intuition.library/WindowLimits

    NAME
        WindowLimits -- Set the minimum and maximum limits of a window.

    SYNOPSIS
        Success = WindowLimits( Window, MinWidth, MinHeight, MaxWidth,
        D0                      A0      D0        D1         D2
                                MaxHeight )
                                D3

        BOOL WindowLimits( struct Window *, WORD, WORD, UWORD, UWORD );

    FUNCTION
        Sets the minimum and maximum limits of the window's size.  Until this
        routine is called, the window's size limits are equal to the initial
        values established in the OpenWindow() function.

        After a call to this routine, the Window will be able to be sized
        to any dimensions within the specified limits.

        If you don't want to change any one of the dimensions, set the limit
        argument for that dimension to zero.  If any of the limit arguments
        is equal to zero, that argument is ignored and the initial setting
        of that parameter remains undisturbed.

        If any of the arguments is out of range (minimums greater than the
        current size, maximums less than the current size), that limit
        will be ignored, though the others will still take effect if they
        are in range.  If any are out of range, the return value from this
        procedure will be FALSE.  If all arguments are valid, the return
        value will be TRUE.

        If you want your window to be able to become "as large as possible"
        you may put -1 (i.e. ~0) in either or both Max arguments.  But
        please note: screen sizes may vary for several reasons, and you
        must be able to handle any possible size of window you might end
        up with if you use this method.  Note that you can use the function
        GetScreenData() to find out how big the screen your window appears in
        is.  That function is particularly useful if your window is in
        the Workbench screen.  You may also refer to the WScreen field
        in your window structure, providing that your window remains open,
        which will ensure that the screen remains open, and thus the
        pointer remains valid.

        If the user is currently sizing this window, the new limits will
        not take effect until after the sizing is completed.

    INPUTS
        Window = pointer to a Window structure
        MinWidth, MinHeight, MaxWidth, MaxHeight = the new limits for the size
            of this window.  If any of these is set to zero, it will
            be ignored and that setting will be unchanged.

    RESULT
        Returns TRUE if everything was in order.  If any of the parameters was
        out of range (minimums greater than current size, maximums less than
        current size), FALSE is returned and the errant limit request is
        not fulfilled (though the valid ones will be).

    BUGS

    SEE ALSO
        GetScreenData()
```

```
intuition.library/WindowToBack                    intuition.library/WindowToBack

    NAME
        WindowToBack -- Ask Intuition to send a window behind others.

    SYNOPSIS
        WindowToBack( Window )
                      A0

        VOID WindowToBack( struct Window * );

    FUNCTION
        This routine sends a request to Intuition asking to send the window
        in back of all other windows in the screen.

        Note that the window will not be depth-arranged immediately, but rather
        will be arranged the next time Intuition receives an input event,
        which happens currently at a minimum rate of ten times per second,
        and a maximum of sixty times a second.

        Remember that WFLG_BACKDROP windows cannot be depth-arranged.

    INPUTS
        Window = pointer to the structure of the window to be sent to the back

    RESULT
        None

    BUGS

    SEE ALSO
        MoveWindow(), SizeWindow(), WindowToFront(), MoveWindowInFrontOf()
```

intuition.library Page 143

```
intuition.library/WindowToFront                   intuition.library/WindowToFront

    NAME
        WindowToFront -- Ask Intuition to bring a window to the front.

    SYNOPSIS
        WindowToFront( Window )
                       A0

        VOID WindowToFront( struct Window * );

    FUNCTION
        This routine sends a request to Intuition asking to bring the window
        in front of all other windows in the screen.

        Note that the window will not be depth-arranged immediately, but rather
        will be arranged the next time Intuition receives an input event,
        which happens currently at a minimum rate of ten times per second,
        and a maximum of sixty times a second.

        Remember that WFLG_BACKDROP windows cannot be depth-arranged.

    INPUTS
        Window = pointer to the structure of the window to be brought to front

    RESULT
        None

    BUGS

    SEE ALSO
        MoveWindow(), SizeWindow(), WindowToBack(), MoveWindowInFrontOf()
```

intuition.library Page 144

```
intuition.library/ZipWindow                           intuition.library/ZipWindow

    NAME
        ZipWindow -- Change window to "alternate" position and
                     dimensions. (V36)

    SYNOPSIS
        ZipWindow( Window )
                   A0

        VOID ZipWindow( struct Window * );

    FUNCTION
        Changes the position and dimension of a window to the values
        at the last occasion of ZipWindow being called (or invoked
        via the "zoom" gadget).

        Typically this is used to snap between a normal, large, working
        dimension of the window to a smaller, more innocuous position
        and dimension.

        Like MoveWindow(), SizeWindow(), and ChangeWindowBox(), the action of
        this function is deferred to the Intuition input handler.

        More tuning needs to be done to establish initial values for
        the first invocation of this function for a window.  You can
        provide initial values using the OpenWindow() tag item
        WA_Zoom.

        It could also use a new name, but "ZoomWindow" is misleading,
        since "Zoom" normally implies "scale."

        The zoom gadget will appear (in the place of the old "toback"
        gadget) when you open your window if you either specify a
        sizing gadget or use WA_Zoom.

        You can detect that this function has taken effect by receiving
        an IDCMP_CHANGEWINDOW IDCMP message.

    INPUTS
        Window -- window to be changed.

    RESULT
        None

    BUGS
        OpenWindow() assumes that the proper default "other" dimensions
        are "full size."

    SEE ALSO
        ChangeWindowBox(), MoveWindow(), SizeWindow()
```

TABLE OF CONTENTS

keymap.library/AskKeyMapDefault
keymap.library/MapANSI
keymap.library/MapRawKey
keymap.library/SetKeyMapDefault

```
keymap.library/AskKeyMapDefault                keymap.library/AskKeyMapDefault

   NAME
        AskKeyMapDefault -- Ask for a pointer to the current default
                            keymap. (V36)

   SYNOPSIS
        keyMap = AskKeyMapDefault()

        struct KeyMap *AskKeyMapDefault();

   FUNCTION
        Return a pointer to the keymap used by the keymap library for
        MapRawKey and MapANSI when a keymap is not specified.

   RESULTS
        keyMap - a pointer to a keyMap structure.  This key map is
            guaranteed to be permanently allocated: it will remain in
            memory till the machine is reset.

   BUGS
        The keymap.h include file should be in the libraries/ or perhaps
        resources/ directory, but is in the devices/ directory for
        compatability reasons.

   SEE ALSO
        devices/keymap.h, keymap.library/SetKeyMapDefault(),
        console.device ...KEYMAP functions
```

```
keymap.library/MapANSI                                keymap.library/MapANSI

   NAME
       MapANSI -- Encode an ANSI string into keycodes. (V36)

   SYNOPSIS
       actual = MapANSI( string, count, buffer, length, keyMap )
       D0                A0      D0     A1      D1      A2

       LONG MapANSI( STRPTR, LONG, STRPTR, LONG, struct KeyMap * );

   FUNCTION
       This console function converts an ANSI byte string to the
       code/qualifier pairs of type IECLASS_RAWKEY that would
       generate the string and places those pairs in a buffer.
       A code/qualifier pair is a pair of bytes suitable for
       putting in the ie_Code and low byte of ie_Qualifier,
       and for subsequent events, shifted to the ie_Prev1DownCode/
       ie_Prev1DownQual then ie_Prev2DownCode/ie_Prev2DownQual
       pairs for any dead or double dead key mapping.


   INPUTS
       string - the ANSI string to convert.
       count - the number of characters in the string.
       buffer - a byte buffer large enough to hold all anticipated
           code/qualifier pairs generated by this conversion.
       length - maximum anticipation, i.e. the buffer size in bytes
           divided by two (the size of the code/qualifier pair).
       keyMap - a KeyMap structure pointer, or null if the default
           key map is to be used.

   RESULT
       actual - the number of code/qualifier pairs in the buffer,
           or negative to describe an error (see below).

   EXAMPLE
       ...
       #include <devices/inputevent.h>

       #define STIMSIZE        3       /* two dead keys, one key */
       unsigned char rBuffer[STIMSIZE*2];
       ...
           KeymapBase = (struct Library *) OpenLibrary("keymap.library", 0);
           ...
           event.ie_NextEvent = 0;
           event.ie_Class = IECLASS_RAWKEY;
           event.ie_SubClass = 0;

           /* prove keymap code completeness and MapANSI reversibility */
           for (code = 0; code < 256; code++) {
               buffer[0] = code;
               actual = MapANSI(buffer, 1, rBuffer, STIMSIZE, 0);
               r = rBuffer;
               event.ie_Prev2DownCode = 0;
               event.ie_Prev2DownQual = 0;
               event.ie_Prev1DownCode = 0;
               event.ie_Prev1DownQual = 0;
               switch (actual) {
                   case -2:
                       printf("MapANSI internal error");
                       goto reportChar;
                   case -1:
                       printf("MapANSI overflow error");
                       goto reportChar;
                   case 0:
```

```
                       printf("MapANSI ungeneratable code");
                       goto reportChar;

                   case 3:
                       event.ie_Prev2DownCode = *r++;
                       event.ie_Prev2DownQual = *r++;
                   case 2:
                       event.ie_Prev1DownCode = *r++;
                       event.ie_Prev1DownQual = *r++;
                   case 1:
                       event.ie_Code = *r++;
                       event.ie_Qualifier = *r;

                       actual = MapRawKey(&event, buffer, BUFFERLEN, 0);
                       if ((actual != 1) || (buffer[0] != code)) {
                           printf("MapANSI not reversible");
                           for (i = 0; i < actual; i++)
                               ReportChar(buffer[i]);
                           printf(" from");
           reportChar:
                           ReportChar(code);
                           printf("\n");
                       }
                   }
               }
           ...

   ERRORS
       if actual is 0, a character in the string was not generatable
           from the keyMap.
       if actual is -1, a buffer overflow condition was detected.
       if actual is -2, an internal error occurred (e.g. no memory)

   SEE ALSO
       devices/inputevent.h, devices/keymap.h
```

```
keymap.library/MapRawKey                                keymap.library/MapRawKey

   NAME
       MapRawKey -- Decode single raw key input event to an ANSI
                    string. (V36)

   SYNOPSIS
       actual = MapRawKey(event, buffer, length, keyMap)
       D0                 A0      A1      D1      A2

       WORD MapRawKey( struct InputEvent *, STRPTR, WORD,
           struct Keymap * );

   FUNCTION
       This console function converts input events of type
       IECLASS_RAWKEY to ANSI bytes, based on the keyMap, and
       places the result into the buffer.

   INPUTS
       event -  an InputEvent structure pointer.  The event list is
           not traversed.
       buffer - a byte buffer large enough to hold all anticipated
           characters generated by this conversion.
       length - maximum anticipation, i.e. the buffer size in bytes.
       keyMap - a KeyMap structure pointer, or null if the default
           key map is to be used.

   RESULT
       actual - the number of characters in the buffer, or -1 if
           a buffer overflow was about to occur.

   EXAMPLE
       ...
       #define BUFFERLEN       80      /* length of longest expected mapping /*
       char buffer[BUFFERLEN];
       struct InputEvent ie;
       ...
           KeymapBase = OpenLibrary("keymap.library", 0);
           ...
           ie.ie_Class = IECLASS_RAWKEY;
           ie.ie_SubClass = 0;
           for (;;) {
               WaitPort(window->UserPort);
               while (im = (struct IntuiMessage *) GetMsg(window->UserPort)) {
                   switch (im->Class) {
                       case RAWKEY:
                           ie.ie_Code = im->Code;
                           ie.ie_Qualifier = im->Qualifier;
                           /* recover dead key codes & qualifiers */
                           ie.ie_EventAddress = (APTR *) *im->IAddress;
                           actual = MapRawKey(&ie, buffer, BUFFERLEN, 0);
                           for (i = 0; i < actual; i++)
                               ReportChar(buffer[i]);
                           break;
                       ...
                   }
                   ...
               }
           }

   ERRORS
       if actual is -1, a buffer overflow condition was detected.
       Not all of the characters in the buffer are valid.

   SEE ALSO
       devices/inputevent.h, devices/keymap.h
```

```
keymap.library/SetKeyMapDefault                  keymap.library/SetKeyMapDefault

   NAME
       SetKeyMapDefault -- Set the current default keymap. (V36)

   SYNOPSIS
       SetKeyMapDefault(keyMap)

       void SetKeyMapDefault( struct KeyMap * );

   FUNCTION
       A pointer to key map specified is cached by the keymap library
       for use by MapRawKey and MapANSI when a keymap is not specified.

   INPUTS
       keyMap - a pointer to a keyMap structure.  This key map must be
           permanently allocated: it must remain in memory till the
           machine is reset.  It is appropriate that this keyMap be a
           node on the keymap.resource list.

   BUGS
       The keymap.h include file should be in the libraries/ or perhaps
       resources/ directory, but is in the devices/ directory for
       compatability reasons.

   SEE ALSO
       devices/keymap.h, keymap.library/AskKeyMapDefault(),
       console.device ...KEYMAP functions
```

TABLE OF CONTENTS

layers.library/BeginUpdate
layers.library/BehindLayer
layers.library/CreateBehindHookLayer
layers.library/CreateBehindLayer
layers.library/CreateUpfrontHookLayer
layers.library/CreateUpfrontLayer
layers.library/DeleteLayer
layers.library/DisposeLayerInfo
layers.library/EndUpdate
layers.library/FattenLayerInfo
layers.library/InitLayers
layers.library/InstallClipRegion
layers.library/InstallLayerHook
layers.library/LockLayer
layers.library/LockLayerInfo
layers.library/LockLayers
layers.library/MoveLayer
layers.library/MoveLayerInFrontOf
layers.library/MoveSizeLayer
layers.library/NewLayerInfo
layers.library/ScrollLayer
layers.library/SizeLayer
layers.library/SwapBitsRastPortClipRect
layers.library/ThinLayerInfo
layers.library/UnlockLayer
layers.library/UnlockLayerInfo
layers.library/UnlockLayers
layers.library/UpfrontLayer
layers.library/WhichLayer

```
layers.library/BeginUpdate                           layers.library/BeginUpdate

   NAME
       BeginUpdate -- Prepare to repair damaged layer.

   SYNOPSIS
       result = BeginUpdate( l )
       d0                    a0

       LONG BeginUpdate( struct Layer *);

   FUNCTION
       Convert damage list to ClipRect list and swap in for
       programmer to redraw through. This routine simulates
       the ROM library environment. The idea is to only render in the
       "damaged" areas, saving time over redrawing all of the layer.
       The layer is locked against changes made by the layer library.

   INPUTS
       l - pointer to a layer

   RESULTS
       result - TRUE if damage list converted to ClipRect list sucessfully.
                FALSE if list conversion aborted. (probably out of memory)

   BUGS
       If BeginUpdate returns FALSE, programmer must abort the attempt to
       refresh this layer and instead call EndUpdate( l, FALSE ) to restore
       original ClipRect and damage list.

   SEE ALSO
       EndUpdate, graphics/layers.h, graphics/clip.h
```

```
layers.library/BehindLayer                              layers.library/BehindLayer

    NAME
        BehindLayer -- Put layer behind other layers.

    SYNOPSIS
        result = BehindLayer( dummy, l )
        d0                    a0     a1

        LONG BehindLayer( LONG, struct Layer *);

    FUNCTION
        Move this layer to the most behind position swapping bits
        in and out of the display with other layers.
        If other layers are REFRESH then collect their damage lists and
        set the LAYERREFRESH bit in the Flags fields of those layers that
        may be revealed.  If this layer is a backdrop layer then
        put this layer behind all other backdrop layers.
        If this layer is NOT a backdrop layer then put in front of the
        top backdrop layer and behind all other layers.

        Note: this operation may generate refresh events in other layers
           associated with this layer's Layer_Info structure.

    INPUTS
        dummy - unused
        l - pointer to a layer

    RESULTS
        result - TRUE    if operation successful
                 FALSE   if operation unsuccessful (probably out of memory)

    BUGS

    SEE ALSO
       graphics/layers.h, graphics/clip.h
```

```
layers.library/CreateBehindHookLayer     layers.library/CreateBehindHookLayer

    NAME                                                                (V36)
        CreateBehindHookLayer -- Create a new layer behind all existing layers,
                                 using supplied callback BackFill hook.

    SYNOPSIS
        result = CreateBehindHookLayer(li,bm,x0,y0,x1,y1,flags,hook,[,bm2])
        d0                             a0 a1 d0 d1 d2 d3 d4    a3   [ a2 ]

        struct Layer *CreateBehindHookLayer(struct Layer_Info *, struct BitMap *,
            LONG, LONG, LONG, LONG, LONG, struct Hook *, ... );

    FUNCTION
        Create a new Layer of position and size (x0,y0)->(x1,y1)
        Make this layer of type found in flags.
        Install Layer->BackFill callback Hook.
        If SuperBitMap, use bm2 as pointer to real SuperBitMap,
        and copy contents of Superbitmap into display layer.
        If this layer is a backdrop layer then place it behind all
        other layers including other backdrop layers. If this is
        not a backdrop layer then place it behind all nonbackdrop
        layers.

        Note: when using SUPERBITMAP, you should also set LAYERSMART flag.

    INPUTS
        li - pointer to LayerInfo structure
        bm - pointer to common BitMap used by all Layers
        x0,y0 - upper left hand corner of layer
        x1,y1 - lower right hand corner of layer
        flags - various types of layers supported as bit sets.
                (for bit definitions, see graphics/layers.h )
        hook -  Layer->BackFill callback Hook which will be called
                with object == (struct RastPort *) result->RastPort
                and message == [ (Layer *) layer, (struct Rectangle) bounds,
                               (WORD) offsetx, (WORD) offsety ]
        bm2 - pointer to optional Super BitMap

    RESULTS
        result - pointer to Layer structure if successful
                 NULL if not successful

    BUGS

    SEE ALSO
        DeleteLayer, graphics/layers.h, graphics/clip.h, graphics/gfx.h
        utility/hooks.h
```

layers.library Page 5

```
layers.library/CreateBehindLayer          layers.library/CreateBehindLayer

   NAME
       CreateBehindLayer -- Create a new layer behind all existing layers.

   SYNOPSIS
       result = CreateBehindLayer(li,bm,x0,y0,x1,y1,flags [,bm2])
       d0                         a0 a1 d0 d1 d2 d3   d4  [ a2 ]

       struct Layer *CreateBehindLayer(struct Layer_Info *, struct BitMap *,
           LONG, LONG, LONG, LONG, LONG, ... );

   FUNCTION
       Create a new Layer of position and size (x0,y0)->(x1,y1)
       Make this layer of type found in flags.
       If SuperBitMap, use bm2 as pointer to real SuperBitMap,
       and copy contents of Superbitmap into display layer.
       If this layer is a backdrop layer then place it behind all
       other layers including other backdrop layers. If this is
       not a backdrop layer then place it behind all nonbackdrop
       layers.

       Note: when using SUPERBITMAP, you should also set LAYERSMART flag.

   INPUTS
       li - pointer to LayerInfo structure
       bm - pointer to common BitMap used by all Layers
       x0,y0 - upper left hand corner of layer
       x1,y1 - lower right hand corner of layer
       flags - various types of layers supported as bit sets.
               (for bit definitions, see graphics/layers.h )
       bm2 - pointer to optional Super BitMap

   RESULTS
       result - pointer to Layer structure if successful
                NULL if not successful

   BUGS

   SEE ALSO
       DeleteLayer, graphics/layers.h, graphics/clip.h, graphics/gfx.h
```

layers.library Page 6

```
layers.library/CreateUpfrontHookLayer   layers.library/CreateUpfrontHookLayer

   NAME                                                           (V36)
       CreateUpfrontHookLayer -- Create a new layer on top of existing layers,
                                 using supplied callback BackFill hook.

   SYNOPSIS
       result = CreateUpfrontHookLayer(li,bm,x0,y0,x1,y1,flags,hook,[,bm2])
       d0                              a0 a1 d0 d1 d2 d3   d4  a3    [ a2 ]

       struct Layer *CreateUpfrontHookLayer(struct Layer_Info *, struct BitMap *,
           LONG, LONG, LONG, LONG, LONG, struct Hook *, ... );

   FUNCTION
       Create a new Layer of position and size (x0,y0)->(x1,y1)
       and place it on top of all other layers.
       Make this layer of type found in flags
       Install Layer->BackFill callback hook.
       if SuperBitMap, use bm2 as pointer to real SuperBitMap.
       and copy contents of Superbitmap into display layer.

       Note: when using SUPERBITMAP, you should also set LAYERSMART flag.

   INPUTS
       li - pointer to LayerInfo structure
       bm - pointer to common BitMap used by all Layers
       x0,y0 - upper left hand corner of layer
       x1,y1 - lower right hand corner of layer
       flags - various types of layers supported as bit sets.
       hook -  Layer->BackFill callback Hook which will be called
               with object == (struct RastPort *) result->RastPort
               and message == [ (Layer *) layer, (struct Rectangle) bounds,
                              (WORD) offsetx, (WORD) offsety ]
       bm2 - pointer to optional Super BitMap

   RESULTS
       result - pointer to Layer structure if successful
                NULL if not successful

   BUGS

   SEE ALSO
      DeleteLayer, graphics/layers.h, graphics/clip.h, graphics/gfx.h
       utility/hooks.h
```

```
layers.library/CreateUpfrontLayer          layers.library/CreateUpfrontLayer

    NAME
        CreateUpfrontLayer -- Create a new layer on top of existing layers.

    SYNOPSIS
        result = CreateUpfrontLayer(li,bm,x0,y0,x1,y1,flags [,bm2])
        d0                          a0 a1 d0 d1 d2 d3   d4  [ a2 ]

        struct Layer *CreateUpfrontLayer(struct Layer_Info *, struct BitMap *,
            LONG, LONG, LONG, LONG, LONG, ... );

    FUNCTION
        Create a new Layer of position and size (x0,y0)->(x1,y1)
        and place it on top of all other layers.
        Make this layer of type found in flags
        if SuperBitMap, use bm2 as pointer to real SuperBitMap.
        and copy contents of Superbitmap into display layer.

        Note: when using SUPERBITMAP, you should also set LAYERSMART flag.

    INPUTS
        li - pointer to LayerInfo structure
        bm - pointer to common BitMap used by all Layers
        x0,y0 - upper left hand corner of layer
        x1,y1 - lower right hand corner of layer
        flags - various types of layers supported as bit sets.
        bm2 - pointer to optional Super BitMap

    RESULTS
        result - pointer to Layer structure if successful
                 NULL if not successful

    BUGS

    SEE ALSO
       DeleteLayer, graphics/layers.h, graphics/clip.h, graphics/gfx.h
```

```
layers.library/DeleteLayer                        layers.library/DeleteLayer

    NAME
        DeleteLayer -- delete layer from layer list.

    SYNOPSTS
        result = DeleteLayer( dummy, l )
        d0                    a0,    a1

        LONG DeleteLayer( LONG, struct Layer *);

    FUNCTION
        Remove this layer from the list of layers.  Release memory
        associated with it.  Restore other layers that may have been
        obscured by it.  Trigger refresh in those that may need it.
        If this is a superbitmap layer make sure SuperBitMap is current.
        The SuperBitMap is not removed from the system but is available
        for program use even though the rest of the layer information has
        been deallocated.

    INPUTS
        dummy - unused
        l - pointer to a layer

    RESULTS
        result - TRUE if this layer successfully deleted from the system
                 FALSE if layer not deleted. (probably out of memory )

    BUGS

    SEE ALSO
        graphics/layers.h, graphics/clip.h
```

```
layers.library/DisposeLayerInfo                layers.library/DisposeLayerInfo

    NAME
        DisposeLayerInfo -- Return all memory for LayerInfo to memory pool

    SYNOPSIS
        DisposeLayerInfo( li )
                          a0

        void DisposeLayerInfo( struct Layer_Info *);

    FUNCTION
        return LayerInfo and any other memory attached to this LayerInfo
        to memory allocator.

        Note: if you wish to delete the layers associated with this Layer_Info
            structure, remember to call DeleteLayer() for each of the layers
            before calling DisposeLayerInfo().

    INPUTS
        li - pointer to LayerInfo structure

    EXAMPLE

        -- delete the layers associated this Layer_Info structure --

        DeleteLayer(li,simple_layer);
        DeleteLayer(li,smart_layer);

        -- see documentation on DeleteLayer about deleting SuperBitMap layers --
        my_super_bitmap_ptr = super_layer->SuperBitMap;
        DeleteLayer(li,super_layer);

        -- now dispose of the Layer_Info structure itself --
        DisposeLayerInfo(li);


    BUGS

    SEE ALSO
        DeleteLayer, graphics/layers.h
```

```
layers.library/EndUpdate                                layers.library/EndUpdate

    NAME
        EndUpdate -- remove damage list and restore state of layer to normal.

    SYNOPSIS
        EndUpdate( l, flag )
                   a0  d0

        void EndUpdate( struct Layer *, UWORD);

    FUNCTION
        After the programmer has redrawn his picture he calls this
        routine to restore the ClipRects to point to his standard
        layer tiling. The layer is then unlocked for access by the
        layer library.

        Note: use flag = FALSE if you are only making a partial update.
            You may use the other region functions (graphics functions such as
            OrRectRegion, AndRectRegion, and XorRectRegion ) to clip adjust
            the DamageList to reflect a partial update.

    INPUTS
        l - pointer to a layer
        flag - use TRUE if update was completed. The damage list is cleared.
               use FALSE if update not complete. The damage list is retained.

    EXAMPLE

        -- begin update for first part of two-part refresh --
        BeginUpdate(my_layer);

        -- do some refresh, but not all --
        my_partial_refresh_routine(my_layer);

        -- end update, false (not completely done refreshing yet) --
        EndUpdate(my_layer, FALSE);

        -- begin update for last part of refresh --
        BeginUpdate(my_layer);

        -- do rest of refresh --
        my_complete_refresh_routine(my_layer);

        -- end update, true (completely done refreshing now) --
        EndUpdate(my_layer, TRUE);

    BUGS

    SEE ALSO
        BeginUpdate, graphics/layers.h, graphics/clip.h
```

```
layers.library/FattenLayerInfo                   layers.library/FattenLayerInfo

    NAME
        FattenLayerInfo -- convert 1.0 LayerInfo to 1.1 LayerInfo
        OBSOLETE OBSOLETE OBSOLETE OBSOLETE OBSOLETE

    SYNOPSIS
        OBSOLETE OBSOLETE OBSOLETE OBSOLETE OBSOLETE
        FattenLayerInfo( li )
                         a0

        LONG FattenLayerInfo( struct Layer_Info *);
        OBSOLETE OBSOLETE OBSOLETE OBSOLETE OBSOLETE

    FUNCTION
        V1.1 software and any later releases need to have more info in the
        Layer_Info structure. To do this in a 1.0 supportable manner requires
        allocation and deallocation of the memory whenever most
        layer library functions are called.  To prevent unnecessary
        allocation/deallocation FattenLayerInfo will preallocate the
        necessary data structures and fake out the layer library into
        thinking it has a LayerInfo gotten from NewLayerInfo.
        NewLayerInfo is the approved method for getting this structure.
        When a program needs to give up the LayerInfo structure it
        must call ThinLayerInfo before freeing the memory. ThinLayerInfo
        is not necessary if New/DisposeLayerInfo are used however.

    INPUTS
        li - pointer to LayerInfo structure

    BUGS

    SEE ALSO

        NewLayerInfo, ThinLayerInfo, DisposeLayerInfo, graphics/layers.h
```

```
layers.library/InitLayers                                layers.library/InitLayers

    NAME
        InitLayers -- Initialize Layer_Info structure
        OBSOLETE OBSOLETE OBSOLETE OBSOLETE OBSOLETE

    SYNOPSIS
        OBSOLETE OBSOLETE OBSOLETE OBSOLETE OBSOLETE
        InitLayers( li )
                    a0

        void InitLayers( struct Layer_Info *);
        OBSOLETE OBSOLETE OBSOLETE OBSOLETE OBSOLETE

    FUNCTION
        Initialize Layer_Info structure in preparation to use
        other layer operations on this list of layers.
        Make the Layers unlocked (open), available to layer operations.

    INPUTS
        li - pointer to LayerInfo structure

    BUGS

    SEE ALSO
        NewLayerInfo, DisposeLayerInfo, graphics/layers.h
```

layers.library Page 13

```
layers.library/InstallClipRegion              layers.library/InstallClipRegion

    NAME
        InstallClipRegion -- Install clip region in layer

    SYNOPSIS
        oldclipregion = InstallClipRegion( l,  region )
        d0                                 a0  a1

        struct Region *InstallClipRegion( struct Layer *, struct Region *);

    FUNCTION
        Installs a transparent Clip region in the layer. All
        subsequent graphics calls will be clipped to this region.
        You MUST remember to call InstallClipRegion(l,NULL) before
        calling DeleteLayer(l) or the Intuition function CloseWindow()
        if you have installed a non-NULL ClipRegion in l.

    INPUTS
        l - pointer to a layer
        region - pointer to a region

    RESULTS
        oldclipregion - The pointer to the previous ClipRegion that
            was installed. Returns NULL if no previous ClipRegion installed.

            Note: If the system runs out of memory while computing the
                resulting ClipRects the LAYERS_CLIPRECTS_LOST bit will
                be set in l->Flags.

    BUGS
        If the system runs out of memory during normal layer operations,
        the ClipRect list may get swept away and not restored.
        As soon as there is enough memory and the layer library
        gets called again the ClipRect list will be rebuilt.

    SEE ALSO
        BeginUpdate EndUpdate,
        graphics/layers.h, graphics/clip.h, graphics/regions.h
```

---

layers.library Page 14

```
layers.library/InstallLayerHook                layers.library/InstallLayerHook

    NAME                                                                (V36)
        InstallLayerHook -- safely install a new Layer->BackFill hook.

    SYNOPSIS
        oldhook = InstallLayerHook( layer, hook )
        d0                          a0     a1

        struct Hook *InstallLayerHook( struct Layer *, struct Hook *);

    FUNCTION
        Installs a new Layer->Backfill Hook, waiting until it is safe to do
        so. Locks the layer while substituting the new Hook and removing the
        old one. If a new Hook is not provided, will install the default layer
        BackFill Hook.

    INPUTS
        layer - pointer to the layer in which to install the Backfill Hook.
        hook -  pointer to layer callback Hook which will be called
                with object == (struct RastPort *) result->RastPort
                and message == [ (Layer *) layer, (struct Rectangle) bounds,
                                 (WORD) offsetx, (WORD) offsety ]

                This hook should fill the Rectangle in the RastPort
                with the BackFill pattern appropriate for offset x/y.

                If this hook pointer is NULL, the function installs
                the "default" Layers BackFill Hook into this Layer.

    RESULTS
        oldhook - pointer to the Layer->BackFill Hook that was previously
                  active.

    BUGS

    SEE ALSO
        graphics/clip.h utility/hooks.h
```

```
layers.library/LockLayer                                    layers.library/LockLayer

    NAME
        LockLayer -- Lock layer to make changes to ClipRects.

    SYNOPSIS
        LockLayer( dummy, l )
                   a0     a1

        void LockLayer( LONG, struct Layer *);

    FUNCTION
        Make this layer unavailable for other tasks to use.
        If another task is already using this layer then wait for
        it to complete and then reserve the layer for your own use.
        (this function does the same thing as graphics.library/LockLayerRom)

        Note: if you wish to lock MORE THAN ONE layer at a time, you
            must call LockLayerInfo() before locking those layers and
            then call UnlockLayerInfo() when you have finished. This
            is to prevent system "deadlocks".

        Further Note: while you hold the lock on a layer, Intuition will block
            on operations such as windowsizing, dragging, menus, and depth
            arranging windows in this layer's screen.  It is recommended that
            YOU do not make Intuition function calls while the layer is locked.

    INPUTS
        dummy - unused
        l - pointer to a layer

    BUGS

    SEE ALSO
        UnlockLayer, LockLayerInfo, UnlockLayerInfo,
        graphics.library/LockLayerRom, graphics/layers.h, graphics/clip.h
```

```
layers.library/LockLayerInfo                            layers.library/LockLayerInfo

    NAME
        LockLayerInfo -- Lock the LayerInfo structure.

    SYNOPSIS
        LockLayerInfo( li )
                       a0

        void LockLayerInfo( struct Layer_Info *);

    FUNCTION
        Before doing an operation that requires the LayerInfo
        structure, make sure that no other task is also using the
        LayerInfo structure.  LockLayerInfo() returns when the
        LayerInfo belongs to this task.  There should be
        an UnlockLayerInfo for every LockLayerInfo.

        Note: All layer routines presently LockLayerInfo() when they
        start up and UnlockLayerInfo() as they exit.  Programmers
        will need to use these Lock/Unlock routines if they wish
        to do something with the LayerStructure that is not
        supported by the layer library.

    INPUTS
        li - pointer to Layer_Info structure

    BUGS

    SEE ALSO
        UnlockLayerInfo, graphics/layers.h
```

```
layers.library/LockLayers                               layers.library/LockLayers

    NAME
        LockLayers -- lock all layers from graphics output.

    SYNOPSIS
        LockLayers( li )
                    a0

        void LockLayers( struct Layer_Info *);

    FUNCTION
        First calls LockLayerInfo()
        Make all layers in this layer list locked.

    INPUTS
        li - pointer to Layer_Info structure

    BUGS

    SEE ALSO
        LockLayer, LockLayerInfo, graphics/layers.h
```

```
layers.library/MoveLayer                                 layers.library/MoveLayer

    NAME
        MoveLayer -- Move layer to new position in BitMap.

    SYNOPSIS
        result = MoveLayer( dummy, l, dx, dy )
        d0                  a0     a1 d0  d1

        LONG MoveLayer( LONG, struct Layer *, LONG, LONG);

    FUNCTION
        Move this layer to new position in shared BitMap.
        If any refresh layers become revealed, collect damage and
        set REFRESH bit in layer Flags.

    INPUTS
        dummy - unused
        l - pointer to a nonbackdrop layer
        dx - delta to add to current x position
        dy - delta to add to current y position

    RETURNS
        result - TRUE if operation successful
                 FALSE if failed (out of memory)

    BUGS
        May not handle (dx,dy) which attempts to move the layer ouside the
        layer's RastPort->BitMap bounds .

    SEE ALSO
        graphics/layers.h, graphics/clip.h
```

```
layers.library/MoveLayerInFrontOf         layers.library/MoveLayerInFrontOf

    NAME
        MoveLayerInFrontOf -- Put layer in front of another layer.

    SYNOPSIS
        result = MoveLayerInFrontOf( layertomove, targetlayer )
                                     a0           a1

        LONG MoveLayerInFrontOf( struct Layer *, struct Layer *);

    FUNCTION
        Move this layer in front of target layer, swapping bits
        in and out of the display with other layers.
        If this is a refresh layer then collect damage list and
        set the LAYERREFRESH bit in layer->Flags if redraw required.

        Note: this operation may generate refresh events in other layers
            associated with this layer's Layer_Info structure.

    INPUTS
        layertomove - pointer to layer which should be moved
        targetlayer - pointer to target layer in front of which to move layer

    RESULTS
        result = TRUE    if operation successful
                 FALSE   if operation unsuccessful (probably out of memory)

    BUGS

    SEE ALSO
        graphics/layers.h
```

```
layers.library/MoveSizeLayer                   layers.library/MoveSizeLayer

    NAME                                                               (V36)
        MoveSizeLayer -- Position/Size layer

    SYNOPSIS
        result = MoveSizeLayer( layer, dx, dy, dw, dh )
        d0                      a0     d0  d1  d2  d3

        LONG MoveSizeLayer( struct Layer *, LONG, LONG, LONG, LONG);

    FUNCTION
        Change upperleft and lower right position of Layer.

    INPUTS
        dummy - unused
        l - pointer to a nonbackdrop layer
        dx,dy - change upper left corner by (dx,dy)
        dw,dy - change size by (dw,dh)

    RETURNS
        result - TRUE if operation successful
                 FALSE if failed (due to out of memory)
                 FALSE if failed (due to illegal layer->bounds)

    BUGS

    SEE ALSO
        graphics/layers.h, graphics/clip.h
```

```
layers.library/NewLayerInfo                          layers.library/NewLayerInfo

    NAME
        NewLayerInfo -- Allocate and Initialize full Layer_Info structure.

    SYNOPSIS
        result = NewLayerInfo()
        d0

        struct Layer_Info *NewLayerInfo( void );

    FUNCTION
        Allocate memory required for full Layer_Info structure.
        Initialize Layer_Info structure in preparation to use
        other layer operations on this list of layers.
        Make the Layer_Info unlocked (open).

    INPUTS
        None

    RESULT
        result- pointer to Layer_Info structure if successful
                NULL if not enough memory

    BUGS

    SEE ALSO
        graphics/layers.h
```

```
layers.library/ScrollLayer                              layers.library/ScrollLayer

    NAME
        ScrollLayer -- Scroll around in a superbitmap, translate coordinates
                       in non-superbitmap layer.

    SYNOPSIS
        ScrollLayer( dummy, l, dx, dy )
                     a0     a1 d0  d1

        void ScrollLayer( LONG, struct Layer *, LONG, LONG);

    FUNCTION
        For a SuperBitMap Layer:
        Update the SuperBitMap from the layer display, then copy bits
        between Layer and SuperBitMap to reposition layer over different
        portion of SuperBitMap.
        For nonSuperBitMap layers, all (x,y) pairs are adjusted by
        the scroll(x,y) value in the layer.  To cause (0,0) to actually
        be drawn at (3,10) use ScrollLayer(-3,-10). This can be useful
        along with InstallClipRegion to simulate Intuition GZZWindows
        without the overhead of an extra layer.

    INPUTS
        dummy - unused
        l - pointer to a layer
        dx - delta to add to current x scroll value
        dy - delta to add to current y scroll value

    BUGS
        May not handle (dx,dy) which attempts to move the layer ouside the
        layer's SuperBitMap bounds.

    SEE ALSO
        graphics/layers.h
```

```
layers.library/SizeLayer                             layers.library/SizeLayer

   NAME
       SizeLayer -- Change the size of this nonbackdrop layer.

   SYNOPSIS
       result = SizeLayer( dummy, l, dx, dy )
       d0                  a0     a1 d0  d1

       LONG SizeLayer( LONG, struct Layer *, LONG, LONG);

   FUNCTION
       Change the size of this layer by (dx,dy). The lower right hand
       corner is extended to make room for the larger layer.
       If there is SuperBitMap for this layer then copy pixels into
       or out of the layer depending on whether the layer increases or
       decreases in size.  Collect damage list for those layers that may
       need to be refreshed if damage occurred.

   INPUTS
       dummy - unused
       l - pointer to a nonbackdrop layer
       dx - delta to add to current x size
       dy - delta to add to current y size

   RESULTS
       result - TRUE if operation successful
                FALSE if failed (out of memory)

   BUGS

   SEE ALSO
       graphics/layers.h, graphics/clip.h
```

```
layers.library/SwapBitsRastPortClipRect layers.library/SwapBitsRastPortClipRect

   NAME
       SwapBitsRastPortClipRect -- Swap bits between common bitmap
                                   and obscured ClipRect

   SYNOPSIS
       SwapBitsRastPortClipRect( rp, cr )
                                 a0  a1

       void SwapBitsRastPortClipRect( struct RastPort *, struct ClipRect *);

   FUNCTION
       Support routine useful for those that need to do some
       operations not done by the layer library.  Allows programmer
       to swap the contents of a small BitMap with a subsection of
       the display. This is accomplished without using extra memory.
       The bits in the display RastPort are exchanged with the
       bits in the ClipRect's BitMap.

       Note: the ClipRect structures which the layer library allocates are
       actually a little bigger than those described in the graphics/clip.h
       include file.  So be warned that it is not a good idea to have
       instances of cliprects in your code.

   INPUTS
       rp - pointer to rastport
       cr - pointer to cliprect to swap bits with

   NOTE
       Because the blit operation started by this function is done asynchronously,
       it is imperative that a WaitBlit() be performed before releasing or using
       the processor to modify any of the associated structures.

   BUGS

   SEE ALSO
       graphics/clip.h, graphics/rastport.h, graphics/clip.h
```

layers.library Page 25

```
layers.library/ThinLayerInfo                    layers.library/ThinLayerInfo

    NAME
        ThinLayerInfo -- convert 1.1 LayerInfo to 1.0 LayerInfo.
        OBSOLETE OBSOLETE OBSOLETE OBSOLETE OBSOLETE

    SYNOPSIS
        OBSOLETE OBSOLETE OBSOLETE OBSOLETE OBSOLETE
        ThinLayerInfo( li )
                       a0

        void ThinLayerInfo( struct Layer_Info *);
        OBSOLETE OBSOLETE OBSOLETE OBSOLETE OBSOLETE

    FUNCTION
        return the extra memory needed that was allocated with
        FattenLayerInfo. This is must be done prior to freeing
        the Layer_Info structure itself. V1.1 software should be
        using DisposeLayerInfo.

    INPUTS
        li - pointer to LayerInfo structure

    BUGS

    SEE ALSO
        DisposeLayerInfo, FattenLayerInfo, graphics/layers.h
```

layers.library Page 26

```
layers.library/UnlockLayer                        layers.library/UnlockLayer

    NAME
        UnlockLayer -- Unlock layer and allow graphics routines to use it.

    SYNOPSIS
        UnlockLayer( l )
                     a0

        void UnlockLayer( struct Layer *);

    FUNCTION
        When finished changing the ClipRects or whatever you were
        doing with this layer you must call UnlockLayer() to allow
        other tasks to proceed with graphic output to the layer.

    INPUTS
        l - pointer to a layer

    BUGS

    SEE ALSO
        graphics/layers.h, graphics/clip.h
```

```
layers.library/UnlockLayerInfo                   layers.library/UnlockLayerInfo

   NAME
       UnlockLayerInfo -- Unlock the LayerInfo structure.

   SYNOPSIS
       UnlockLayerInfo( li )
                        a0

       void UnlockLayerInfo( struct Layer_Info *);

   FUNCTION
       After the operation is complete that required a LockLayerInfo,
       unlock the LayerInfo structure so that  other tasks may
       affect the layers.

   INPUTS
       li - pointer to the Layer_Info structure

    BUGS

    SEE ALSO
       LockLayerInfo, graphics/layers.h
```

```
layers.library/UnlockLayers                            layers.library/UnlockLayers

   NAME
       UnlockLayers -- Unlock all layers from graphics output.
                       Restart graphics output to layers that have been waiting

   SYNOPSIS
       UnlockLayers( li )
                     a0

       void UnlockLayers( struct Layer_Info *);

   FUNCTION
       Make all layers in this layer list unlocked.
       Then call UnlockLayerInfo

   INPUTS
       li - pointer to the Layer_Info structure

   BUGS

   SEE ALSO
       LockLayers, UnlockLayer, graphics/layers.h
```

layers.library Page 29

```
layers.library/UpfrontLayer                          layers.library/UpfrontLayer

    NAME
        UpfrontLayer -- Put layer in front of all other layers.

    SYNOPSIS
        result = UpfrontLayer( dummy, l )
        d0                     a0     a1

        LONG UpfrontLayer( LONG, struct Layer *);

    FUNCTION
        Move this layer to the most upfront position swapping bits
        in and out of the display with other layers.
        If this is a refresh layer then collect damage list and
        set the LAYERREFRESH bit in layer->Flags if redraw required.
        By clearing the BACKDROP bit in the layers Flags you may
        bring a Backdrop layer up to the front of all other layers.

        Note: this operation may generate refresh events in other layers
           associated with this layer's Layer_Info structure.

    INPUTS
        dummy - unused
        l - pointer to a nonbackdrop layer

    RESULTS
        result - TRUE   if operation successful
                 FALSE  if operation unsuccessful (probably out of memory)

    BUGS

    SEE ALSO
        graphics/layers.h
```

layers.library Page 30

```
layers.library/WhichLayer                                  layers.library/WhichLayer

    NAME
        WhichLayer -- Which Layer is this point in?

    SYNOPSIS
        layer = WhichLayer( li, x, y )
        d0                  a0  d0 d1

        struct Layer *WhichLayer(struct Layer_Info*, WORD, WORD);

    FUNCTION
        Starting at the topmost layer check to see if this point (x,y)
            occurs in this layer.  If it does return the pointer to this
            layer. Return NULL if there is no layer at this point.

    INPUTS
        li  = pointer to LayerInfo structure
        (x,y) = coordinate in the BitMap

    RESULTS
        layer - pointer to the topmost layer that this point is in
                NULL if this point is not in a layer

    SEE ALSO
        graphics/layers.h
```

TABLE OF CONTENTS

mathieeedoubbas.library/IEEEDPAbs
mathieeedoubbas.library/IEEEDPAdd
mathieeedoubbas.library/IEEEDPCeil
mathieeedoubbas.library/IEEEDPCmp
mathieeedoubbas.library/IEEEDPDiv
mathieeedoubbas.library/IEEEDPFix
mathieeedoubbas.library/IEEEDPFloor
mathieeedoubbas.library/IEEEDPFlt
mathieeedoubbas.library/IEEEDPMul
mathieeedoubbas.library/IEEEDPNeg
mathieeedoubbas.library/IEEEDPSub
mathieeedoubbas.library/IEEEDPTst

```
mathieeedoubbas.library/IEEEDPAbs          mathieeedoubbas.library/IEEEDPAbs

   NAME
        IEEEDPAbs -- compute absolute value of IEEE double precision argument

   SYNOPSIS
          x    = IEEEDPAbs(  y  );
        d0/d1              d0/d1

        double  x,y;

   FUNCTION
        Take the absolute value of argument y and return it to caller.

   INPUTS
        y -- IEEE double precision floating point value

   RESULT
        x -- IEEE double precision floating point value

   BUGS

   SEE ALSO
```

mathieeedoubbas.library Page 3

```
mathieeedoubbas.library/IEEEDPAdd          mathieeedoubbas.library/IEEEDPAdd

   NAME
        IEEEDPAdd -- add one double precision IEEE number to another

   SYNOPSIS
          x   = IEEEDPAdd(  y  ,  z  );
        d0/d1             d0/d1 d2/d3

        double  x,y,z;

   FUNCTION
        Compute x = y + z in IEEE double precision.

   INPUTS
        y -- IEEE double precision floating point value
        z -- IEEE double precision floating point value

   RESULT
        x -- IEEE double precision floating point value

   BUGS

   SEE ALSO
        IEEEDPSub
```

mathieeedoubbas.library Page 4

```
mathieeedoubbas.library/IEEEDPCeil          mathieeedoubbas.library/IEEEDPCeil

   NAME
        IEEEDPCeil -- compute Ceil function of IEEE double precision number

   SYNOPSIS
          x   = IEEEDPCeil(  y  );
        d0/d1              d0/d1

        double  x,y;

   FUNCTION
        Calculate the least integer greater than or equal to x and return it.
        This value may have more than 32 bits of significance.
        This identity is true.  Ceil(x) = -Floor(-x).

   INPUTS
        y -- IEEE double precision floating point value

   RESULT
        x -- IEEE double precision floating point value

   BUGS

   SEE ALSO
        IEEEDPFloor
```

```
mathieeedoubbas.library/IEEEDPCmp          mathieeedoubbas.library/IEEEDPCmp

   NAME
       IEEEDPCmp -- compare two double precision floating point numbers

   SYNOPSIS
         c    = IEEEDPCmp(  y  ,  z  );
        d0                 d0/d1 d2/d3

       double  y,z;
       long    c;

   FUNCTION
       Compare y with z. Set the condition codes for less, greater, or
       equal. Set return value c to -1 if y<z, or +1 if y>z, or 0 if
       y == z.

   INPUTS
       y -- IEEE double precision floating point value
       z -- IEEE double precision floating point value

   RESULT
      c = 1    cc = gt         for (y > z)
      c = 0    cc = eq         for (y == z)
      c = -1   cc = lt         for (y < z)

   BUGS

   SEE ALSO
```

```
mathieeedoubbas.library/IEEEDPDiv          mathieeedoubbas.library/IEEEDPDiv

   NAME
       IEEEDPDiv -- divide one double precision IEEE by another

   SYNOPSIS
         x    = IEEEDPDiv(  y  ,  z  );
       d0/d1               d0/d1 d2/d3

       double  x,y,z;

   FUNCTION
       Compute x = y / z in IEEE double precision.

   INPUTS
       y -- IEEE double precision floating point value
       z -- IEEE double precision floating point value

   RESULT
       x -- IEEE double precision floating point value

   BUGS

   SEE ALSO
       IEEEDPMul
```

mathieeedoubbas.library Page 7

```
mathieeedoubbas.library/IEEEDPFix          mathieeedoubbas.library/IEEEDPFix

   NAME
       IEEEDPFix -- convert IEEE double float to integer

   SYNOPSIS
       x    = IEEEDPFix(  y  );
       d0               d0/d1

       long    x;
       double  y;

   FUNCTION
       Convert IEEE double precision argument to a 32 bit signed integer
       and return result.

   INPUTS
       y -- IEEE double precision floating point value

   RESULT
       if no overflow occured then return
               x -- 32 bit signed integer
       if overflow return largest +- integer
               For round to zero

   BUGS

   SEE ALSO
       IEEEDPFlt
```

mathieeedoubbas.library Page 8

```
mathieeedoubbas.library/IEEEDPFloor        mathieeedoubbas.library/IEEEDPFloor

   NAME
       IEEEDPFloor -- compute Floor function of IEEE double precision number

   SYNOPSIS
         x    = IEEEDPFloor(  y  );
       d0/d1                d0/d1

       double  x,y;

   FUNCTION
       Calculate the largest integer less than or equal to x and return it.
       This value may have more than 32 bits of significance.

   INPUTS
       y -- IEEE double precision floating point value

   RESULT
       x -- IEEE double precision floating point value

   BUGS

   SEE ALSO
       IEEEDPCeil
```

mathieeedoubbas.library Page 9

```
mathieeedoubbas.library/IEEEDPFlt          mathieeedoubbas.library/IEEEDPFlt

   NAME
        IEEEDPFlt -- convert integer to IEEE double precision number

   SYNOPSIS
          x    = IEEEDPFlt(  y  );
        d0/d1                d0

        double  x;
        long    y;

   FUNCTION
        Convert a signed 32 bit value to a double precision IEEE value
        and return it in d0/d1. No exceptions can occur with this
        function.

   INPUTS
        y -- 32 bit integer in d0

   RESULT
        x is a 64 bit double precision IEEE value

   BUGS

   SEE ALSO
        IEEEDPFix
```

mathieeedoubbas.library Page 10

```
mathieeedoubbas.library/IEEEDPMul          mathieeedoubbas.library/IEEEDPMul

   NAME
        IEEEDPMul -- multiply one double precision IEEE number by another

   SYNOPSIS
          x    = IEEEDPMul(  y   ,   z  );
        d0/d1              d0/d1  d2/d3

        double  x,y,z;

   FUNCTION
        Compute x = y * z in IEEE double precision.

   INPUTS
        y -- IEEE double precision floating point value
        z -- IEEE double precision floating point value

   RESULT
        x -- IEEE double precision floating point value

   BUGS

   SEE ALSO
        IEEEDPDiv
```

mathieeedoubbas.library Page 11

```
mathieeedoubbas.library/IEEEDPNeg          mathieeedoubbas.library/IEEEDPNeg

   NAME
       IEEEDPNeg -- compute negative value of IEEE double precision number

   SYNOPSIS
         x    = IEEEDPNeg(  y  );
       d0/d1              d0/d1

       double  x,y;

   FUNCTION
       Invert the sign of argument y and return it to caller.

   INPUTS
       y - IEEE double precision floating point value

   RESULT
       x - IEEE double precision floating point value

   BUGS

   SEE ALSO
```

mathieeedoubbas.library Page 12

```
mathieeedoubbas.library/IEEEDPSub          mathieeedoubbas.library/IEEEDPSub

   NAME
       IEEEDPSub -- subtract one double precision IEEE number from another

   SYNOPSIS
         x    = IEEEDPSub(  y  ,  z  );
       d0/d1              d0/d1 d2/d3

       double  x,y,z;

   FUNCTION
       Compute x = y - z in IEEE double precision.

   INPUTS
       y -- IEEE double precision floating point value
       z -- IEEE double precision floating point value

   RESULT
       x -- IEEE double precision floating point value

   BUGS

   SEE ALSO
       IEEEDPAdd
```

```
mathieeedoubbas.library/IEEEDPTst          mathieeedoubbas.library/IEEEDPTst

   NAME
        IEEEDPTst -- compare IEEE double precision value to 0.0

   SYNOPSIS
          c   = IEEEDPTst(  y  );
          d0                d0/d1

        double  y;
        long    c;

   FUNCTION
        Compare y to 0.0, set the condition codes for less than, greater
        than, or equal to 0.0.  Set the return value c to -1 if less than,
        to +1 if greater than, or 0 if equal to 0.0.

   INPUTS
        y -- IEEE double precision floating point value

   RESULT
        c = 1   cc = gt         for (y > 0.0)
        c = 0   cc = eq         for (y == 0.0)
        c = -1  cc = lt         for (y < 0.0)

   BUGS

   SEE ALSO
```

mathieeedoubtrans.library Page 1

TABLE OF CONTENTS

mathieeedoubtrans.library/IEEEDPAcos
mathieeedoubtrans.library/IEEEDPAsin
mathieeedoubtrans.library/IEEEDPAtan
mathieeedoubtrans.library/IEEEDPCos
mathieeedoubtrans.library/IEEEDPCosh
mathieeedoubtrans.library/IEEEDPExp
mathieeedoubtrans.library/IEEEDPFieee
mathieeedoubtrans.library/IEEEDPLog
mathieeedoubtrans.library/IEEEDPLog10
mathieeedoubtrans.library/IEEEDPPow
mathieeedoubtrans.library/IEEEDPSin
mathieeedoubtrans.library/IEEEDPSincos
mathieeedoubtrans.library/IEEEDPSinh
mathieeedoubtrans.library/IEEEDPSqrt
mathieeedoubtrans.library/IEEEDPTan
mathieeedoubtrans.library/IEEEDPTanh
mathieeedoubtrans.library/IEEEDPTieee

mathieeedoubtrans.library Page 2

```
mathieeedoubtrans.library/IEEEDPAcos     mathieeedoubtrans.library/IEEEDPAcos

   NAME
        IEEEDPAcos -- compute the arc cosine of a number

   SYNOPSIS
          x   = IEEEDPAcos(  y  );
        d0/d1              d0/d1

        double  x,y;

   FUNCTION
        Compute arc cosine of y in IEEE double precision

   INPUTS
        y - IEEE double precision floating point value

   RESULT
        x - IEEE double precision floating point value

   BUGS

   SEE ALSO
        IEEEDPCos(), IEEEDPAtan(), IEEEDPAsin()
```

mathieeedoubtrans.library Page 3

```
mathieeedoubtrans.library/IEEEDPAsin     mathieeedoubtrans.library/IEEEDPAsin

   NAME
        IEEEDPAsin -- compute the arcsine of a number

   SYNOPSIS
          x   = IEEEDPAsin(  y  );
        d0/d1                d0/d1

        double  x,y;

   FUNCTION
        Compute the arc sine of y in IEEE double precision

   INPUTS
        y - IEEE double precision floating point value

   RESULT
        x - IEEE double precision floating point value

   BUGS

   SEE ALSO
        IEEEDPSin(), IEEEDPAtan(), IEEEDPAcos()
```

mathieeedoubtrans.library Page 4

```
mathieeedoubtrans.library/IEEEDPAtan     mathieeedoubtrans.library/IEEEDPAtan

   NAME
        IEEEDPAtan -- compute the arctangent of a floating point number

   SYNOPSIS
          x   = IEEEDPAtan(  y  );
        d0/d1                d0/d1

        double  x,y;

   FUNCTION
        Compute arctangent of y in IEEE double precision

   INPUTS
        y - IEEE double precision floating point value

   RESULT
        x - IEEE double precision floating point value

   BUGS

   SEE ALSO
        IEEEDPTan(), IEEEDPAsin(), IEEEDPACos()
```

mathieeedoubtrans.library Page 5

```
mathieeedoubtrans.library/IEEEDPCos       mathieeedoubtrans.library/IEEEDPCos

   NAME
        IEEEDPCos -- compute the cosine of a floating point number

   SYNOPSIS
          x   = IEEEDPCos(  y  );
        d0/d1               d0/d1

        double  x,y;

   FUNCTION
        Compute cosine of y in IEEE double precision

   INPUTS
        y - IEEE double precision floating point value

   RESULT
        x - IEEE double precision floating point value

   BUGS

   SEE ALSO
        IEEEDPAcos(), IEEEDPSin(), IEEEDPTan()
```

mathieeedoubtrans.library Page 6

```
mathieeedoubtrans.library/IEEEDPCosh     mathieeedoubtrans.library/IEEEDPCosh

   NAME
        IEEEDPCosh -- compute the hyperbolic cosine of a floating point number

   SYNOPSIS
          x   = IEEEDPCosh(  y  );
        d0/d1                d0/d1

        double  x,y;

   FUNCTION
        Compute hyperbolic cosine of y in IEEE double precision

   INPUTS
        y - IEEE double precision floating point value

   RESULT
        x - IEEE double precision floating point value

   BUGS

   SEE ALSO
        IEEEDPSinh(), IEEEDPTanh()
```

```
mathieeedoubtrans.library/IEEEDPExp       mathieeedoubtrans.library/IEEEDPExp

   NAME
        IEEEDPExp -- compute the exponential of e

   SYNOPSIS
          x    = IEEEDPExp(  y  );
        d0/d1               d0/d1

        double  x,y;

   FUNCTION
        Compute e^y in IEEE double precision

   INPUTS
        y - IEEE double precision floating point value

   RESULT
        x - IEEE double precision floating point value

   BUGS

   SEE ALSO
        IEEEDPLog()
```

```
mathieeedoubtrans.library/IEEEDPFieee   mathieeedoubtrans.library/IEEEDPFieee

   NAME
        IEEEDPFieee -- convert IEEE single to IEEE double

   SYNOPSIS
          x    = IEEEDPFieee(  y  );
        d0/d1                  d0

        float   y;
        double  x;

   FUNCTION
        Convert IEEE single precision number to IEEE double precision.

   INPUTS
        y - IEEE single precision floating point value

   RESULT
        x - IEEE double precision floating point value

   BUGS

   SEE ALSO
        IEEEDPTieee()
```

mathieeedoubtrans.library Page 9

```
mathieeedoubtrans.library/IEEEDPLog          mathieeedoubtrans.library/IEEEDPLog

   NAME
       IEEEDPLog -- compute the natural logarithm of a floating point number

   SYNOPSIS
         x   = IEEEDPLog(  y  );
       d0/d1              d0/d1

       double  x,y;

   FUNCTION
       Compute ln(y) in IEEE double precision

   INPUTS
       y - IEEE double precision floating point value

   RESULT
       x - IEEE double precision floating point value

   BUGS

   SEE ALSO
       IEEEDPExp()
```

mathieeedoubtrans.library Page 10

```
mathieeedoubtrans.library/IEEEDPLog10   mathieeedoubtrans.library/IEEEDPLog10

   NAME
       IEEEDPLog10 -- compute logarithm base 10 of a number

   SYNOPSIS
         x   = IEEEDPLog10(  y  );
       d0/d1                d0/d1

       double  x,y;

   FUNCTION
       Compute the logarithm base 10 of y in IEEE double precision

   INPUTS
       y - IEEE double precision floating point value

   RESULT
       x - IEEE double precision floating point value

   BUGS

   SEE ALSO
       IEEEDPLog()
```

mathieeedoubtrans.library Page 11

```
mathieeedoubtrans.library/IEEEDPPow       mathieeedoubtrans.library/IEEEDPPow

   NAME
        IEEEDPPow -- raise a number to another number power

   SYNOPSIS
          z   = IEEEDPPow(  x  ,  y  );
        d0/d1              d2/d3 d0/d1

        double  x,y,z;

   FUNCTION
        Compute y^x in IEEE double precision

   INPUTS
        x - IEEE double precision floating point value
        y - IEEE double precision floating point value

   RESULT
        z - IEEE double precision floating point value

   BUGS

   SEE ALSO
```

---

mathieeedoubtrans.library Page 12

```
mathieeedoubtrans.library/IEEEDPSin       mathieeedoubtrans.library/IEEEDPSin

   NAME
        IEEEDPSin -- compute the sine of a floating point number

   SYNOPSIS
          x   = IEEEDPSin(  y  );
        d0/d1              d0/d1

        double  x,y;

   FUNCTION
        Compute sine of y in IEEE double precision

   INPUTS
        y - IEEE double precision floating point value

   RESULT
        x - IEEE double precision floating point value

   BUGS

   SEE ALSO
        IEEEDPAsin(), IEEEDPTan(), IEEEDPCos()
```

mathieeedoubtrans.library Page 13

```
mathieeedoubtrans.library/IEEEDPSincos mathieeedoubtrans.library/IEEEDPSincos

   NAME
       IEEEDPSincos -- compute the arc tangent of a floating point number

   SYNOPSIS
         x    = IEEEDPSincos( z ,  y  );
       d0/d1                 a0  d0/d1

       double  x,y,*z;

   FUNCTION
       Compute sin and cosine of y in IEEE double precision.
       Store the cosine in *z. Return the sine of y.

   INPUTS
       y - IEEE double precision floating point value
       z - pointer to IEEE double precision floating point number

   RESULT
       x - IEEE double precision floating point value

   BUGS

   SEE ALSO
       IEEEDPSin(), IEEEDPCos()
```

mathieeedoubtrans.library Page 14

```
mathieeedoubtrans.library/IEEEDPSinh      mathieeedoubtrans.library/IEEEDPSinh

   NAME
       IEEEDPSinh -- compute the hyperbolic sine of a floating point number

   SYNOPSIS
         x    = IEEEDPSinh(  y  );
       d0/d1               d0/d1

       double  x,y;

   FUNCTION
       Compute hyperbolic sine of y in IEEE double precision

   INPUTS
       y - IEEE double precision floating point value

   RESULT
       x - IEEE double precision floating point value

   BUGS

   SEE ALSO
       IEEEDPCosh, IEEEDPTanh
```

```
mathieeedoubtrans.library/IEEEDPSqrt     mathieeedoubtrans.library/IEEEDPSqrt

   NAME
       IEEEDPSqrt -- compute the square root of a number

   SYNOPSIS
         x    = IEEEDPSqrt(  y  );
       d0/d1                d0/d1

       double  x,y;

   FUNCTION
       Compute square root of y in IEEE double precision

   INPUTS
       y - IEEE double precision floating point value

   RESULT
       x - IEEE double precision floating point value

   BUGS

   SEE ALSO
```

```
mathieeedoubtrans.library/IEEEDPTan       mathieeedoubtrans.library/IEEEDPTan

   NAME
       IEEEDPTan -- compute the tangent of a floating point number

   SYNOPSIS
         x    = IEEEDPTan(  y  );
       d0/d1                d0/d1

       double  x,y;

   FUNCTION
       Compute tangent of y in IEEE double precision

   INPUTS
       y - IEEE double precision floating point value

   RESULT
       x - IEEE double precision floating point value

   BUGS

   SEE ALSO
       IEEEDPAtan(), IEEEDPSin(), IEEEDPCos()
```

mathieeedoubtrans.library Page 17

```
mathieeedoubtrans.library/IEEEDPTanh     mathieeedoubtrans.library/IEEEDPTanh

   NAME
        IEEEDPTanh -- compute the hyperbolic tangent of a floating point number

   SYNOPSIS
          x   = IEEEDPTanh(  y  );
        d0/d1              d0/d1

        double  x,y;

   FUNCTION
        Compute hyperbolic tangent of y in IEEE double precision

   INPUTS
        y - IEEE double precision floating point value

   RESULT
        x - IEEE double precision floating point value

   BUGS

   SEE ALSO
        IEEEDPSinh(), IEEEDPCosh()
```

---

mathieeedoubtrans.library Page 18

```
mathieeedoubtrans.library/IEEEDPTieee   mathieeedoubtrans.library/IEEEDPTieee

   NAME
        IEEEDPTieee -- convert IEEE double to IEEE single

   SYNOPSIS
          x   = IEEEDPTieee(  y  );
         d0                 d0/d1

        double  y;
        float   x;

   FUNCTION
        Convert IEEE double precision number to IEEE single precision.

   INPUTS
        y - IEEE double precision floating point value

   RESULT
        x - IEEE single precision floating point value

   BUGS

   SEE ALSO
        IEEEDPFieee()
```

TABLE OF CONTENTS

mathieeesingbas.library/IEEESPAbs
mathieeesingbas.library/IEEESPAdd
mathieeesingbas.library/IEEESPCeil
mathieeesingbas.library/IEEESPCmp
mathieeesingbas.library/IEEESPDiv
mathieeesingbas.library/IEEESPFix
mathieeesingbas.library/IEEESPFloor
mathieeesingbas.library/IEEESPFlt
mathieeesingbas.library/IEEESPMul
mathieeesingbas.library/IEEESPNeg
mathieeesingbas.library/IEEESPSub
mathieeesingbas.library/IEEESPTst

```
mathieeesingbas.library/IEEESPAbs           mathieeesingbas.library/IEEESPAbs

   NAME
       IEEESPAbs -- compute absolute value of IEEE single precision argument

   SYNOPSIS
         x   = IEEESPAbs(  y  )
        d0                d0

       float IEEESPAbs(float);

   FUNCTION
       Take the absolute value of argument y and return it to caller.

   INPUTS
       y -- IEEE single precision floating point value

   RESULT
       x -- IEEE single precision floating point value

   BUGS

   SEE ALSO
```

matheieeesingbas.library Page 3

```
mathieeesingbas.library/IEEESPAdd            mathieeesingbas.library/IEEESPAdd

   NAME
        IEEESPAdd -- add one single precision IEEE number to another

   SYNOPSIS
          x   = IEEESPAdd(  y  ,  z  )
         d0                d0    d1

          float IEEESPAdd(float,float);

   FUNCTION
        Compute x = y + z in IEEE single precision.

   INPUTS
        y -- IEEE single precision floating point value
        z -- IEEE single precision floating point value

   RESULT
        x -- IEEE single precision floating point value

   BUGS

   SEE ALSO
        IEEESPSub()
```

matheieeesingbas.library Page 4

```
mathieeesingbas.library/IEEESPCeil          mathieeesingbas.library/IEEESPCeil

   NAME
        IEEESPCeil -- compute Ceil function of IEEE single precision number

   SYNOPSIS
          x   = IEEESPCeil(  y  )
         d0                 d0

          float IEEESPCeil(float);

   FUNCTION
        Calculate the least integer greater than or equal to x and return it.
        This identity is true.  Ceil(x) = -Floor(-x).

   INPUTS
        y -- IEEE single precision floating point value

   RESULT
        x -- IEEE single precision floating point value

   BUGS

   SEE ALSO
        IEEESPFloor()
```

matheieeesingbas.library Page 5

```
mathieeesingbas.library/IEEESPCmp          mathieeesingbas.library/IEEESPCmp

   NAME
      IEEESPCmp -- compare two single precision floating point numbers

   SYNOPSIS
         c    = IEEESPCmp(  y  ,   z  )
         d0                 d0    d1

         long IEEESPCmp(float,float);

   FUNCTION
      Compare y with z. Set the condition codes for less, greater, or
      equal. Set return value c to -1 if y<z, or +1 if y>z, or 0 if
      y == z.

   INPUTS
      y -- IEEE single precision floating point value
      z -- IEEE single precision floating point value

   RESULT
     c = 1    cc = gt         for (y > z)
     c = 0    cc = eq         for (y == z)
     c = -1   cc = lt         for (y < z)

   BUGS

   SEE ALSO
```

---

matheieeesingbas.library Page 6

```
mathieeesingbas.library/IEEESPDiv          mathieeesingbas.library/IEEESPDiv

   NAME
      IEEESPDiv -- divide one single precision IEEE by another

   SYNOPSIS
         x    = IEEESPDiv(  y  ,   z  )
        d0                  d0    d1

         float IEEESPDiv(float,float);

   FUNCTION
      Compute x = y / z in IEEE single precision.
      Note that the Motorola fast floating point Div routine reverses
      the order of the arguments for the C interface, although the
      dividend is still in d0 and the divisor is in d1.

   INPUTS
      y -- IEEE single precision floating point value
      z -- IEEE single precision floating point value

   RESULT
      x -- IEEE single precision floating point value

   BUGS

   SEE ALSO
      IEEESPMul()
```

matheieeesingbas.library Page 7

```
mathieeesingbas.library/IEEESPFix             mathieeesingbas.library/IEEESPFix

   NAME
        IEEESPFix -- convert IEEE single float to integer

   SYNOPSIS
        x    = IEEESPFix(  y  )
        d0                 d0

        long IEEESPFix(float);

   FUNCTION
        Convert IEEE single precision argument to a 32 bit signed integer
        and return result.

   INPUTS
        y -- IEEE single precision floating point value

   RESULT
        if no overflow occured then return
                x -- 32 bit signed integer
        if overflow return largest +- integer
                For round to zero

   BUGS

   SEE ALSO
        IEEESPFlt()
```

matheieeesingbas.library Page 8

```
mathieeesingbas.library/IEEESPFloor         mathieeesingbas.library/IEEESPFloor

   NAME
        IEEESPFloor -- compute Floor function of IEEE single precision number

   SYNOPSIS
          x    = IEEESPFloor(  y  )
          d0                   d0

          float IEEESPFloor(float);

   FUNCTION
        Calculate the largest integer less than or equal to x and return it.

   INPUTS
        y -- IEEE single precision floating point value

   RESULT
        x -- IEEE single precision floating point value

   BUGS

   SEE ALSO
        IEEESPCeil()
```

```
mathieeesingbas.library/IEEESPFlt           mathieeesingbas.library/IEEESPFlt

   NAME
        IEEESPFlt -- convert integer to IEEE single precision number

   SYNOPSIS
          x    = IEEESPFlt(  y  )
         d0                 d0

           float IEEESPFlt(long);

   FUNCTION
        Convert a signed 32 bit value to a single precision IEEE value
        and return it in d0. No exceptions can occur with this
        function.

   INPUTS
        y -- 32 bit integer in d0

   RESULT
        x is a 32 bit single precision IEEE value

   BUGS

   SEE ALSO
        IEEESPFix()
```

```
mathieeesingbas.library/IEEESPMul           mathieeesingbas.library/IEEESPMul

   NAME
        IEEESPMul -- multiply one double precision IEEE number by another

   SYNOPSIS
          x    = IEEESPMul(  y  ,  z  )
         d0                 d0     d1

           float IEEESPMul(float,float);

   FUNCTION
        Compute x = y * z in IEEE single precision.

   INPUTS
        y -- IEEE single precision floating point value
        z -- IEEE single precision floating point value

   RESULT
        x -- IEEE single precision floating point value

   BUGS

   SEE ALSO
        IEEESPDiv()
```

matheieeesingbas.library Page 11

```
mathieeesingbas.library/IEEESPNeg          mathieeesingbas.library/IEEESPNeg

   NAME
        IEEESPNeg -- compute negative value of IEEE single precision number

   SYNOPSIS
          x   = IEEESPNeg(  y  )
          d0                d0

          float IEEESPNeg(float);

   FUNCTION
        Invert the sign of argument y and return it to caller.

   INPUTS
        y - IEEE single precision floating point value

   RESULT
        x - IEEE single precision floating point value

   BUGS

   SEE ALSO
```

matheieeesingbas.library Page 12

```
mathieeesingbas.library/IEEESPSub          mathieeesingbas.library/IEEESPSub

   NAME
        IEEESPSub -- subtract one single precision IEEE number from another

   SYNOPSIS
          x   = IEEESPSub(  y  ,  z  )
         d0                d0     d1

          float IEEESPSub(float,float);

   FUNCTION
        Compute x = y - z in IEEE single precision.

   INPUTS
        y -- IEEE single precision floating point value
        z -- IEEE single precision floating point value

   RESULT
        x -- IEEE single precision floating point value

   BUGS

   SEE ALSO
        IEEESPAdd()
```

```
mathieeesingbas.library/IEEESPTst         mathieeesingbas.library/IEEESPTst

   NAME
        IEEESPTst -- compare IEEE single precision value to 0.0

   SYNOPSIS
          c    = IEEESPTst(  y  )
          d0                 d0

          long IEEESPTst(float);

   FUNCTION
        Compare y to 0.0, set the condition codes for less than, greater
        than, or equal to 0.0.  Set the return value c to -1 if less than,
        to +1 if greater than, or 0 if equal to 0.0.

   INPUTS
        y -- IEEE single precision floating point value

   RESULT
        c = 1    cc = gt          for (y > 0.0)
        c = 0    cc = eq          for (y == 0.0)
        c = -1   cc = lt          for (y < 0.0)

   BUGS

   SEE ALSO
```

matheieeesingtrans.library Page 1

TABLE OF CONTENTS

mathieeesingtrans.library/IEEESPAcos
mathieeesingtrans.library/IEEESPAsin
mathieeesingtrans.library/IEEESPAtan
mathieeesingtrans.library/IEEESPCos
mathieeesingtrans.library/IEEESPCosh
mathieeesingtrans.library/IEEESPExp
mathieeesingtrans.library/IEEESPFieee
mathieeesingtrans.library/IEEESPLog
mathieeesingtrans.library/IEEESPLog10
mathieeesingtrans.library/IEEESPPow
mathieeesingtrans.library/IEEESPSin
mathieeesingtrans.library/IEEESPSincos
mathieeesingtrans.library/IEEESPSinh
mathieeesingtrans.library/IEEESPSqrt
mathieeesingtrans.library/IEEESPTan
mathieeesingtrans.library/IEEESPTanh
mathieeesingtrans.library/IEEESPTieee

matheieeesingtrans.library Page 2

```
mathieeesingtrans.library/IEEESPAcos     mathieeesingtrans.library/IEEESPAcos

   NAME
        IEEESPAcos -- compute the arc cosine of a number

   SYNOPSIS
          x    = IEEESPAcos(  y  );
        d0                   d0

        float   x,y;

   FUNCTION
        Compute arc cosine of y in IEEE single precision

   INPUTS
        y - IEEE single precision floating point value

   RESULT
        x - IEEE single precision floating point value

   BUGS

   SEE ALSO
        IEEESPCos(), IEEESPAtan(), IEEESPAsin()
```

matheieeesingtrans.library Page 3

```
mathieeesingtrans.library/IEEESPAsin     mathieeesingtrans.library/IEEESPAsin

   NAME
        IEEESPAsin -- compute the arcsine of a number

   SYNOPSIS
          x    = IEEESPAsin(  y   );
        d0                  d0

        float   x,y;

   FUNCTION
        Compute the arc sine of y in IEEE single precision

   INPUTS
        y - IEEE single precision floating point value

   RESULT
        x - IEEE single precision floating point value

   BUGS

   SEE ALSO
        IEEESPSin(), IEEESPAtan(), IEEESPAcos()
```

matheieeesingtrans.library Page 4

```
mathieeesingtrans.library/IEEESPAtan     mathieeesingtrans.library/IEEESPAtan

   NAME
        IEEESPAtan -- compute the arc tangent of number

   SYNOPSIS
          x    = IEEESPAtan(  y  );
        d0                  d0

        single  x,y;

   FUNCTION
        Compute arctangent of y in IEEE single precisi n

   INPUTS
        y - IEEE single precision floating point value

   RESULT
        x - IEEE single precision floating point value

   BUGS

   SEE ALSO
```

matheieeesingtrans.library Page 5

```
mathieeesingtrans.library/IEEESPCos       mathieeesingtrans.library/IEEESPCos

   NAME
        IEEESPCos -- compute the cosine of a floating point number

   SYNOPSIS
          x    = IEEESPCos(  y  );
        d0                  d0

        float   x,y;

   FUNCTION
        Compute cosine of y in IEEE single precision

   INPUTS
        y - IEEE single precision floating point value

   RESULT
        x - IEEE single precision floating point value

   BUGS

   SEE ALSO
        IEEESPAcos(), IEEESPSin(), IEEESPTan()
```

matheieeesingtrans.library Page 6

```
mathieeesingtrans.library/IEEESPCosh     mathieeesingtrans.library/IEEESPCosh

   NAME
        IEEESPCosh -- compute the hyperbolic cosine of a floating point number

   SYNOPSIS
          x    = IEEESPCosh(  y  );
        d0                  d0

        float   x,y;

   FUNCTION
        Compute hyperbolic cosine of y in IEEE single precision

   INPUTS
        y - IEEE single precision floating point value

   RESULT
        x - IEEE single precision floating point value

   BUGS

   SEE ALSO
        IEEESPSinh(), IEEESPTanh()
```

matheieeesingtrans.library Page 7

```
mathieeesingtrans.library/IEEESPExp       mathieeesingtrans.library/IEEESPExp

   NAME
        IEEESPExp -- compute the exponential of e

   SYNOPSIS
          x    = IEEESPExp(  y  );
        d0                  d0

        float   x,y;

   FUNCTION
        Compute e^y in IEEE single precision

   INPUTS
        y - IEEE single precision floating point value

   RESULT
        x - IEEE single precision floating point value

   BUGS

   SEE ALSO
        IEEESPLog()
```

matheieeesingtrans.library Page 8

```
mathieeesingtrans.library/IEEESPFieee   mathieeesingtrans.library/IEEESPFieee

   NAME
        IEEESPFieee -- convert IEEE single to IEEE single

   SYNOPSIS
          x    = IEEESPFieee(  y  );
        d0                    d0

        float   y;
        float  x;

   FUNCTION
        Convert IEEE single precision number to IEEE single precision.
        These are included for completeness although they just
        return the input parameter. A good way to remember how these
        functions work is: They convert to and from the local format
        to Single Precision IEEE. The local format for this library
        happens to also be Single Precision IEEE.

   INPUTS
        y - IEEE single precision floating point value

   RESULT
        x - IEEE single precision floating point value

   BUGS

   SEE ALSO
        IEEESPTieee()
```

matheieeesingtrans.library Page 9

```
mathieeesingtrans.library/IEEESPLog       mathieeesingtrans.library/IEEESPLog

   NAME
        IEEESPLog -- compute the natural logarithm of a floating point number

   SYNOPSIS
          x    = IEEESPLog(  y  );
        d0                 d0

        float   x,y;

   FUNCTION
        Compute ln(y) in IEEE single precision

   INPUTS
        y - IEEE single precision floating point value

   RESULT
        x - IEEE single precision floating point value

   BUGS

   SEE ALSO
        IEEESPExp()
```

matheieeesingtrans.library Page 10

```
mathieeesingtrans.library/IEEESPLog10   mathieeesingtrans.library/IEEESPLog10

   NAME
        IEEESPLog10 -- compute logarithm base 10 of a number

   SYNOPSIS
          x    = IEEESPLog10(  y  );
        d0                   d0

        float   x,y;

   FUNCTION
        Compute the logarithm base 10 of y in IEEE single precision

   INPUTS
        y - IEEE single precision floating point value

   RESULT
        x - IEEE single precision floating point value

   BUGS

   SEE ALSO
        IEEESPLog()
```

```
mathieeesingtrans.library/IEEESPPow        mathieeesingtrans.library/IEEESPPow

   NAME
       IEEESPPow -- raise a number to another number power

   SYNOPSIS
         z    = IEEESPPow(  x  ,  y  );
       d0                  d1 d0

       float   x,y,z;

   FUNCTION
       Compute y^x in IEEE single precision

   INPUTS
       x - IEEE single precision floating point value
       y - IEEE single precision floating point value

   RESULT
       z - IEEE single precision floating point value

   BUGS

   SEE ALSO
```

```
mathieeesingtrans.library/IEEESPSin        mathieeesingtrans.library/IEEESPSin

   NAME
       IEEESPSin -- compute the sine of a floating point number

   SYNOPSIS
         x    = IEEESPSin(  y  );
       d0                  d0

       float   x,y;

   FUNCTION
       Compute sine of y in IEEE single precision

   INPUTS
       y - IEEE single precision floating point value

   RESULT
       x - IEEE single precision floating point value

   BUGS

   SEE ALSO
       IEEESPAsin(), IEEESPTan(), IEEESPCos()
```

```
mathieeesingtrans.library/IEEESPSincos mathieeesingtrans.library/IEEESPSincos

   NAME
        IEEESPSincos -- compute the arc tangent of a floating point number

   SYNOPSIS
          x    = IEEESPSincos( z ,  y  );
        d0                     a0  d0

        float   x,y,*z;

   FUNCTION
        Compute sin and cosine of y in IEEE single precision.
        Store the cosine in *z. Return the sine of y.

   INPUTS
        y - IEEE single precision floating point value
        z - pointer to IEEE single precision floating point number

   RESULT
        x - IEEE single precision floating point value

   BUGS

   SEE ALSO
        IEEESPSin(), IEEESPCos()
```

```
mathieeesingtrans.library/IEEESPSinh     mathieeesingtrans.library/IEEESPSinh

   NAME
        IEEESPSinh -- compute the hyperbolic sine of a floating point number

   SYNOPSIS
          x    = IEEESPSinh(  y  );
        d0                   d0

        float   x,y;

   FUNCTION
        Compute hyperbolic sine of y in IEEE single precision

   INPUTS
        y - IEEE single precision floating point value

   RESULT
        x - IEEE single precision floating point value

   BUGS

   SEE ALSO
        IEEESPCosh, IEEESPTanh
```

```
mathieeesingtrans.library/IEEESPSqrt     mathieeesingtrans.library/IEEESPSqrt

   NAME
        IEEESPSqrt -- compute the square root of a number

   SYNOPSIS
          x   = IEEESPSqrt(  y  );
         d0                 d0

        float   x,y;

   FUNCTION
        Compute square root of y in IEEE single precision

   INPUTS
        y - IEEE single precision floating point value

   RESULT
        x - IEEE single precision floating point value

   BUGS

   SEE ALSO
```

```
mathieeesingtrans.library/IEEESPTan       mathieeesingtrans.library/IEEESPTan

   NAME
        IEEESPTan -- compute the tangent of a floating point number

   SYNOPSIS
          x   = IEEESPTan(  y  );
         d0                d0

        float   x,y;

   FUNCTION
        Compute tangent of y in IEEE single precision

   INPUTS
        y - IEEE single precision floating point value

   RESULT
        x - IEEE single precision floating point value

   BUGS

   SEE ALSO
        IEEESPAtan(), IEEESPSin(), IEEESPCos()
```

```
mathieeesingtrans.library/IEEESPTanh     mathieeesingtrans.library/IEEESPTanh

   NAME
       IEEESPTanh -- compute the hyperbolic tangent of a floating point number

   SYNOPSIS
         x   = IEEESPTanh(  y  );
       d0                 d0

       float   x,y;

   FUNCTION
       Compute hyperbolic tangent of y in IEEE single precision

   INPUTS
       y - IEEE single precision floating point value

   RESULT
       x - IEEE single precision floating point value

   BUGS

   SEE ALSO
       IEEESPSinh(), IEEESPCosh()
```

```
mathieeesingtrans.library/IEEESPTieee   mathieeesingtrans.library/IEEESPTieee

   NAME
       IEEESPTieee -- convert IEEE single to IEEE single

   SYNOPSIS
         x   = IEEESPTieee(  y  );
        d0                 d0

       float   y;
       float   x;

   FUNCTION
       Convert IEEE single precision number to IEEE single precision.
       These are included for completeness although they just
       return the input parameter. A good way to remember how these
       functions work is: They convert to and from the local format
       to Single Precision IEEE. The local format for this library
       happens to also be Single Precision IEEE.

   INPUTS
       y - IEEE single precision floating point value

   RESULT
       x - IEEE single precision floating point value

   BUGS

   SEE ALSO
       IEEESPFieee()
```

TABLE OF CONTENTS

mathtrans.library/SPAcos
mathtrans.library/SPAsin
mathtrans.library/SPAtan
mathtrans.library/SPCos
mathtrans.library/SPCosh
mathtrans.library/SPExp
mathtrans.library/SPFieee
mathtrans.library/SPLog
mathtrans.library/SPLog10
mathtrans.library/SPPow
mathtrans.library/SPSin
mathtrans.library/SPSincos
mathtrans.library/SPSinh
mathtrans.library/SPSqrt
mathtrans.library/SPTan
mathtrans.library/SPTanh
mathtrans.library/SPTieee

mathtrans.library/SPAcos　　mathtrans.library/SPAcos

NAME

SPAcos - obtain the arccosine of the floating point number

SYNOPSIS

```
fnum2 = SPAcos(fnum1);
               d0.1
float fnum2;
float fnum1;
```

FUNCTION

Accepts a floating point number representing the cosine of an angle and returns the value of said angle in radians

INPUTS

fnum1 - Motorola fast floating point number

RESULT

fnum2 - Motorola fast floating point number

BUGS

None

SEE ALSO

SPSin

mathtrans.library Page 3

```
mathtrans.library/SPAsin                               mathtrans.library/SPAsin

NAME

        SPAsin - obtain the arcsine of the floating point number

SYNOPSIS

        fnum2 = SPAsin(fnum1);
                       d0.1
        float fnum2;
        float fnum1;

FUNCTION

        Accepts a floating point number representing the sine
        of an angle and returns the value of said angle in
        radians

INPUTS

        fnum1 - Motorola fast floating point number

RESULT

        fnum2 - Motorola fast floating point number

BUGS

        None

SEE ALSO

        SPCos
```

---

mathtrans.library Page 4

```
mathtrans.library/SPAtan                               mathtrans.library/SPAtan

NAME

        SPAtan - obtain the arctangent of the floating point number

SYNOPSIS

        fnum2 = SPAtan(fnum1);
                       d0.1
        float fnum2;
        float fnum1;

FUNCTION

        Accepts a floating point number representing the tangent
        of an angle and returns the value of said angle in
        radians

INPUTS

        fnum1 - Motorola fast floating point number

RESULT

        fnum2 - Motorola fast floating point number

BUGS

        None

SEE ALSO

        SPTan
```

mathtrans.library

Page 5

```
mathtrans.library/SPCos                           mathtrans.library/SPCos

NAME

       SPCos - obtain the cosine of the floating point number

SYNOPSIS

       fnum2 = SPCos(fnum1);
                     d0.1
       float fnum2;
       float fnum1;

FUNCTION

       Accepts a floating point number representing an angle
       in radians and returns the cosine of said angle.

INPUTS

       fnum1 - Motorola fast floating point number

RESULT

       fnum2 - Motorola fast floating point number

BUGS

       None

SEE ALSO

       SPAcos
```

mathtrans.library

Page 6

```
mathtrans.library/SPCosh                          mathtrans.library/SPCosh

NAME

       SPCosh - obtain the hyperbolic cosine of the floating point number

SYNOPSIS

       fnum2 = SPCosh(fnum1);
                      d0.1
       float fnum2;
       float fnum1;

FUNCTION

       Accepts a floating point number representing an angle
       in radians and returns the hyperbolic cosine of said angle.

INPUTS

       fnum1 - Motorola fast floating point number

RESULT

       fnum2 - Motorola fast floating point number

BUGS

       None

SEE ALSO

       SPSinh
```

mathtrans.library Page 7

```
mathtrans.library/SPExp                                   mathtrans.library/SPExp

NAME

        SPExp - obtain the exponential (e**X) of the floating point number

SYNOPSIS

        fnum2 = SPExp(fnum1);
                       d0.1
        float fnum2;
        float fnum1;

FUNCTION

        Accepts a floating point number and returns the value
        of e raised to the fnum1 power

INPUTS

        fnum1 - Motorola fast floating point number

RESULT

        fnum2 - Motorola fast floating point number

BUGS

        None

SEE ALSO

        SPLog
```

mathtrans.library Page 8

```
mathtrans.library/SPFieee                                 mathtrans.library/SPFieee

NAME

        SPFieee - convert single precision ieee to FFP number

SYNOPSIS

        fnum = SPFieee(ieeenum);
                       d0.1
        float fnum;
        float ieeenum;

FUNCTION

        Accepts a standard single precision format
        returns the same number, converted to Motorola
        fast floating point number

INPUTS

        ieeenum - IEEE Single Precision Floating Point

RESULT

        fnum - Motorola fast floating point number

BUGS

        None

SEE ALSO

        SPTieee
```

mathtrans.library Page 9

```
mathtrans.library/SPLog                                  mathtrans.library/SPLog

NAME

        SPLog - obtain the natural logarithm of the floating point number

SYNOPSIS

        fnum2 = SPLog(fnum1);
                      d0.1
        float fnum2;
        float fnum1;

FUNCTION

        Accepts a floating point number and returns the natural
        logarithem (base e) of said number

INPUTS

        fnum1 - Motorola fast floating point number

RESULT

        fnum2 - Motorola fast floating point number

BUGS

        None

SEE ALSO

        SPExp
```

---

mathtrans.library Page 10

```
mathtrans.library/SPLog10                              mathtrans.library/SPLog10

NAME

        SPLog10 - obtain the naperian logarithm(base 10) of the
                  floating point number

SYNOPSIS

        fnum2 = SPLog10(fnum1);
                        d0.1
        float fnum2;
        float fnum1;

FUNCTION

        Accepts a floating point number and returns the naperian
        logarithm (base 10) of said number

INPUTS

        fnum1 - Motorola fast floating point number

RESULT

        fnum2 - Motorola fast floating point number

BUGS

        None

SEE ALSO

        SPExp, SpLog
```

mathtrans.library Page 11

```
mathtrans.library/SPPow                                mathtrans.library/SPPow

NAME

        SPPow - raise a number to a power

SYNOPSIS

        result = SPPow(fnum1, fnum2);
                       d1.1    d0.1
        float fnum1, fnum2;
        float result;

FUNCTION

        Accepts two floating point numbers and returns the
        result of fnum2 raised to the fnum1 power

INPUTS

        fnum1 - Motorola fast floating point number
        fnum2 - Motorola fast floating point number

RESULT

        result - Motorola fast floating point number

BUGS

        None

SEE ALSO

        SPExp, SPLog
```

mathtrans.library Page 12

```
mathtrans.library/SPSin                                mathtrans.library/SPSin

NAME

        SPSin - obtain the sine of the floating point number

SYNOPSIS

        fnum2 = SPSin(fnum1);
                      d0.1
        float fnum2;
        float fnum1;

FUNCTION

        Accepts a floating point number representing an angle
        in radians and returns the sine of said angle.

INPUTS

        fnum1 - Motorola fast floating point number

RESULT

        fnum2 - Motorola fast floating point number

BUGS

        None

SEE ALSO

        SPAsin
```

```
mathtrans.library/SPSincos                               mathtrans.library/SPSincos

NAME

        SPSincos - obtain the sine and cosine of a number

SYNOPSIS

        fnum3 = SPSincos(pfnum2, fnum1);
                         d1.1,   d0.1
        float *pfnum2;
        float fnum1;
        float fnum3;

FUNCTION

        Accepts a floating point number (fnum1) representing
        an angle in radians and a pointer to another floating
        point number (pfnum2). It computes the cosine and places it in
        *pfnum2. It computes the sine and returns it as a result.

INPUTS

        fnum1 - Motorola fast floating point number
        pfnum2 - pointer to Motorola fast floating point number

RESULT

        *pfnum2 - Motorola fast floating point number (cosine)
        fnum3 - Motorola fast floating point number (sine)

BUGS

        None

SEE ALSO

        SPSin, SPCos
```

```
mathtrans.library/SPSinh                                       mathtrans.library/SPSinh

NAME

        SPSinh - obtain the hyperbolic sine of the floating point number

SYNOPSIS

        fnum2 = SPSinh(fnum1);
                       d0.1
        float fnum2;
        float fnum1;

FUNCTION

        Accepts a floating point number representing an angle
        in radians and returns the hyperbolic sine of said angle.

INPUTS

        fnum1 - Motorola fast floating point number

RESULT

        fnum2 - Motorola fast floating point number

BUGS

        None

SEE ALSO

        SPCosh
```

mathtrans.library Page 15

```
mathtrans.library/SPSqrt                              mathtrans.library/SPSqrt

NAME

        SPSqrt - obtain the square root of the floating point number

SYNOPSIS

        fnum2 = SPSqrt(fnum1);
                       d0.1
        float fnum2;
        float fnum1;

FUNCTION

        Accepts a floating point number and returns the square toot
        of said number

INPUTS

        fnum1 - Motorola fast floating point number

RESULT

        fnum2 - Motorola fast floating point number

BUGS

        None

SEE ALSO

        SPPow, SPMul
```

mathtrans.library Page 16

```
mathtrans.library/SPTan                               mathtrans.library/SPTan

NAME

        SPTan - obtain the tangent of the floating point number

SYNOPSIS

        fnum2 = SPTan(fnum1);
                      d0.1
        float fnum2;
        float fnum1;

FUNCTION

        Accepts a floating point number representing an angle
        in radians and returns the tangent of said angle.

INPUTS

        fnum1 - Motorola fast floating point number

RESULT

        fnum2 - Motorola fast floating point number

BUGS

        None

SEE ALSO

        SPAtan
```

```
mathtrans.library/SPTanh                              mathtrans.library/SPTanh

NAME

        SPTanh - obtain the hyperbolic tangent of the floating point number

SYNOPSIS

        fnum2 = SPTanh(fnum1);
                       d0.1
        float fnum2;
        float fnum1;

FUNCTION

        Accepts a floating point number representing an angle
        in radians and returns the hyperbolic tangent of said angle.

INPUTS

        fnum1 - Motorola fast floating point number

RESULT

        fnum2 - Motorola fast floating point number

BUGS

        None

SEE ALSO

        SPSinh, SPCosh
```

```
mathtrans.library/SPTieee                              mathtrans.library/SPTieee

NAME

        SPTieee - convert FFP number to single precision ieee

SYNOPSIS

        ieeenum = SPTieee(fnum);
                          d0.1
        float ieeenum;
        float fnum;

FUNCTION

       Accepts a Motorola fast floating point number and
       returns the same number, converted into IEEE
       standard single precision format

INPUTS

        fnum - Motorola fast floating point number

RESULT

        ieeenum - IEEE Single Precision Floating Point

BUGS

        None

SEE ALSO

        SPFieee
```

TABLE OF CONTENTS

rexxsyslib.library/ClearRexxMsg
rexxsyslib.library/CreateArgstring
rexxsyslib.library/CreateRexxMsg
rexxsyslib.library/DeleteArgstring
rexxsyslib.library/DeleteRexxMsg
rexxsyslib.library/FillRexxMsg
rexxsyslib.library/IsRexxMsg
rexxsyslib.library/LengthArgstring
rexxsyslib.library/LockRexxBase
rexxsyslib.library/UnlockRexxBase

```
rexxsyslib.library/ClearRexxMsg                     rexxsyslib.library/ClearRexxMsg

    NAME
        ClearRexxMsg - Releases and clears the argument array in a RexxMsg

    SYNOPSIS
        ClearRexxMsg(msgptr, count)
                     A0      D0

        VOID ClearRexxMsg(struct RexxMsg *,ULONG)

    FUNCTION
        This function will DeleteArgstring() one or more argstrings from
        the RexxMsg and clear the slot.  The count is used to select the
        number of slots to clear.

    INPUTS
        msgptr - A pointer to a RexxMsg
        count - The number of slots to be cleared.  The number can be from
                1 to 16.  (There are 16 slots)

    RESULTS
        All of the slots in the given count will be cleared and the argstring
        will have been released.

    SEE ALSO
        FillRexxMsg(), DeleteRexxMsg(), DeleteArgstring(), CreateArgstring()

    BUGS
```

rexxsyslib.library Page 3

```
rexxsyslib.library/CreateArgstring        rexxsyslib.library/CreateArgstring

    NAME
        CreateArgstring - Create an argument string structure

    SYNOPSIS
        argstr = CreateArgstring(string, length)
        D0,A0                    A0      D0

        UBYTE *CreateArgstring(UBYTE *, ULONG);

    FUNCTION
        Allocates a RexxArg structure and copies the supplied string into it.
        The returned pointer points at the string part of the structure
        and can be treated like an ordinary string pointer.  (However, care
        must be taken that you do not change the string)

    INPUTS
        string - A pointer at your input string
        length - The number of bytes of your input string you wish copied.
                 (NOTE:  You are limited to 65,535 byte strings)

    RESULTS
        argstr - A pointer to the argument string.  The results are returned
                 in both A0 and D0.  You should always check the result
                 as an allocation failure would cause an error.

    SEE ALSO
        DeleteArgstring(), LengthArgstring(), ClearRessMsg(), FillRexxMsg()

    BUGS
```

rexxsyslib.library Page 4

```
rexxsyslib.library/CreateRexxMsg            rexxsyslib.library/CreateRexxMsg

    NAME
        CreateRexxMsg - Create an ARexx message structure

    SYNOPSIS
        rexxmsg = CreateRexxMsg(port, extension, host)
        D0,A0                   A0    A1         A2

        struct RexxMsg *CreateRexxMsg(struct MsgPort *, UBYTE *, UBYTE *)

    FUNCTION
        This functions allocates an ARexx message packet.  The RexxMsg
        consists of a standard EXEC message structure extended to include
        the ARexx specific information.

    INPUTS
        port - A pointer to a public or private message port.  This *MUST*
               be a valid port as this is where the message will be replied.

        extension - A pointer to a NULL terminated string that is to be used
                    as the default extension for the REXX scripts.  If this
                    is NULL, the default is "REXX"

        host - A pointer to a NULL terminated string that is to be used
               as the default host port.  The name must be the same as the
               name of the public message port that is to be the default host.
               If this field is NULL, the default is REXX.

    RESULTS
        rexxmsg - A RexxMsg structure

    NOTES
        The extension and host strings must remain valid for as long as the
        RexxMsg exists as only the pointer to those strings are stored.

    SEE ALSO
        DeleteRexxMsg(), ClearRexxMsg(), FillRexxMsg()

    BUGS
```

rexxsyslib.library Page 5

```
rexxsyslib.library/DeleteArgstring          rexxsyslib.library/DeleteArgstring

    NAME
        DeleteArgstring - Releases an Argstring created by CreateArgstring()

    SYNOPSIS
        DeleteArgstring(argstring)
                        A0

        VOID DeleteArgstring(UBYTE *)

    FUNCTION
        Releases an argstring.  The argstring must have been created by ARexx

    INPUTS
        argstring - A pointer to the string buffer of an argstring.

    RESULTS

    SEE ALSO
        CreateArgstring(), ClearRexxMsg(), FillRexxMsg()

    BUGS
```

rexxsyslib.library Page 6

```
rexxsyslib.library/DeleteRexxMsg              rexxsyslib.library/DeleteRexxMsg

    NAME
        DeleteRexxMsg - Releases a RexxMsg structure created by CreateRexxMsg()

    SYNOPSIS
        DeleteRexxMsg(packet)
                      A0

        VOID DeleteRexxMsg(struct RexxMsg *)

    FUNCTION
        The function releases an ARexx message packet that was allocated
        with CreateRexxMsg().  Any argument fields in the RexxMsg structure
        should be cleared before calling this function as it does
        not release them for you.

    INPUTS
        packet - A pointer to a RexxMsg structure allocated by CreateRexxMsg()

    EXAMPLE
        if (rmsg=CreateRexxMsg(myport,"myapp","MYAPP_PORT"))
        {
                /* Do my think with rmsg */
                ClearRexxMsg(rmsg,16);  /* We may not want to clear all 16 */
                DeleteRexxMsg(rmsg);
        }

    SEE ALSO
        CreateRexxMsg(), ClearRexxMsg()

    BUGS
```

```
rexxsyslib.library/FillRexxMsg                    rexxsyslib.library/FillRexxMsg

    NAME
        FillRexxMsg - Fill the argument strings as needed

    SYNOPSIS
        result = FillRexxMsg(msgptr, count, mask)
        D0                   A0      D0     D1 [0:15]

        BOOL FillRexxMsg(struct RexxMsg *,ULONG,ULONG)

    FUNCTION
        This function will convert and install up to 16 argument strings into
        a RexxMsg structure.  The message packet's argument fields must be
        set to either a pointer to a NULL terminated string or an integer value
        The mask, bits 0 to 15, correspond to the type of value is stored
        in the argument slot.  If the bit is cleared, the argument is a
        string pointer; if the bit is cleared, the argument is an integer.

    INPUTS
        msgptr - Pointer to a RexxMsg (allocated via CreateRexxMsg)
        count - The number of argument slots to fill in.  This number should
                be from 1 to 16.

        mask - A bit mask corresponding to the 16 fields that is used to
               determine the type of the field.

    RESULTS
        result - A boolean.  If it is TRUE, the call worked.  If it is false,
                 some allocation did not work.  All argstrings that were
                 created will be released.

    SEE ALSO
        ClearRexxMsg(), CreateArgstring(), DeleteArgstring(), CreateRexxMsg()

    BUGS
```

```
rexxsyslib.library/IsRexxMsg                        rexxsyslib.library/IsRexxMsg

    NAME
        IsRexxMsg - Function to determine if a message came from ARexx

    SYNOPSIS
        result = IsRexxMsg(msgptr)
        D0                 A0

        BOOL IsRexxMsg(struct RexxMsg *)

    FUNCTION
        This function can be used to determine if a message came from an
        ARexx program.

    INPUTS
        msgptr - A pointer to the suspected RexxMsg.

    RESULTS
        result - A boolean:  TRUE if it is an ARexx message, FALSE if not.

    SEE ALSO
        CreateRexxMsg()

    BUGS
```

rexxsyslib.library

```
rexxsyslib.library/LengthArgstring         rexxsyslib.library/LengthArgstring

    NAME
        LengthArgstring - Returns the length value stored in the argstring

    SYNOPSIS
        length = LengthArgstring(argstring)
        D0                       A0

        ULONG LengthArgstring(UBYTE *)

    FUNCTION
        This function returns the length value stored in the argstring.
        This is *NOT* the same as doing a strlen() type call on the
        argstring.  (Note that argstrings may contain NULLs)

    INPUTS
        argstring - A pointer to an argstring that was created by ARexx

    RESULTS
        length - The length of the argstring.

    EXAMPLE

    SEE ALSO
        CreateArgstring()

    BUGS
```

rexxsyslib.library

```
rexxsyslib.library/LockRexxBase               rexxsyslib.library/LockRexxBase

    NAME
        LockRexxBase - Obtain a semaphore lock on the RexxBase structure

    SYNOPSIS
        LockRexxBase(resource)
                     D0

        VOID LockRexxBase(ULONG)

    FUNCTION
        Secures the specified resource in the ARexx library base.

    INPUTS
        resource - A manifest constant defining which resource to lock.
                   ZERO locks all resources.

    NOTES
        Currently, only ZERO resource type is available.  You *MUST* make
        sure that you do not call this function with an undefined value
        as it may become defined at some future date and cause unwanted
        behavior.

    SEE ALSO
        UnlockRexxBase()

    BUGS
```

```
rexxsyslib.library/UnlockRexxBase          rexxsyslib.library/UnlockRexxBase

    NAME
        UnlockRexxBase - Release a semaphore lock on the RexxBase structure

    SYNOPSIS
        UnlockRexxBase(resource)
                       D0

        VOID UnlockRexxBase(ULONG)

    FUNCTION
        Releases the specified resource in the ARexx library base.

    INPUTS
        resource - A manifest constant defining which resource to unlock.
                   This value *MUST* match the value used in the matching
                   LockRexxBase() call.

    NOTES
        Currently, only ZERO resource type is available.  You *MUST* make
        sure that you do not call this function with an undefined value
        as it may become defined at some future date and cause unwanted
        behavior.  You *MUST* make sure that you only call this function
        after a matching call to LockRexxBase() was made.

    SEE ALSO
        LockRexxBase()

    BUGS
```

translator.library

Page 1

```
TABLE OF CONTENTS

translator.library/Translate
```

translator.library

Page 2

```
translator.library/Translate                          translator.library/Translate

    NAME
        Translate -- Convert an English string into narrator device phonemes.

    SYNOPSIS
        rtnCode = Translate(inString, inLength, outBuffer, outLength)
        D0                  A0        D0        A1         D1

        LONG Translate( STRPTR inString, LONG inLength, STRPTR outBuffer,
            LONG outlen );

    FUNCTION
        The translate function converts an English string into
        a string of phonetic codes suitable as input to the
        narrator device.

    INPUTS
        inString - pointer to English string
        inLength - length of English string
        outBuffer - a char array which will hold the phonetic codes
        outLength - the length of the output array

    RESULTS
        rtnCode - zero if no error has occured.
            The only error that can occur is overflowing the outBuffer.
            If Translate() determines that an overflow will occur, it
            will stop the translation at a word boundary before the
            overflow happens.  If this occurs, rtnCode will be a
            negative number whose absolute value indicates where in
            inString Translate() stopped.  The user can then use the
            offset -rtnCode from the beginning of inString in a
            subsequent Translate() call to continue the translation.

    SEE ALSO
        narrator.device/CMD_WRITE
```

TABLE OF CONTENTS

utility.library/AllocateTagItems
utility.library/Amiga2Date
utility.library/CallHookPkt
utility.library/CheckDate
utility.library/CloneTagItems
utility.library/Date2Amiga
utility.library/FilterTagChanges
utility.library/FilterTagItems
utility.library/FindTagItem
utility.library/FreeTagItems
utility.library/GetTagData
utility.library/MapTags
utility.library/NextTagItem
utility.library/PackBoolTags
utility.library/RefreshTagItemClones
utility.library/SDivMod32
utility.library/SMult32
utility.library/Stricmp
utility.library/Strnicmp
utility.library/TagInArray
utility.library/ToLower
utility.library/ToUpper
utility.library/UDivMod32
utility.library/UMult32

```
utility.library/AllocateTagItems              utility.library/AllocateTagItems

   NAME
       AllocateTagItems --  Allocate a TagItem array (or chain). (V36)

   SYNOPSIS
       tagList = AllocateTagItems( numItems )
       D0                          D0

       struct TagItem *AllocateTagItems( ULONG numItems);

   FUNCTION
       Allocates the specified number of usable TagItems slots, and does
       so in a format that the function FreeTagItems can handle.

       Note that to access the TagItems in 'tagList', you should use
       the function NextTagItem().  This will insure you respect any
       chaining (TAG_MORE) that the list uses, and will skip any
       TAG_IGNORE items that AllocateTagItems() might use to stash
       size and other information.

   INPUTS
       numItems        - the number of TagItem slots you want to allocate.

   RESULT
       tagList         - the allocated chain of TagItem structures.  Will
                         return NULL if unsuccessful.

   BUGS

   SEE ALSO
       FreeTagItems(), CloneTagItems()
```

utility.library Page 3

```
utility.library/Amiga2Date                          utility.library/Amiga2Date

   NAME
       Amiga2Date -- Calculate the date from a timestamp.  (V36)

   SYNOPSIS
       Amiga2Date( AmigaTime, Date )
                   D0         A0

       void Amiga2Date( ULONG, struct ClockData * );

   FUNCTION
       Fills in a ClockData structure with the date and time calculated
       from a ULONG containing the number of seconds from 01-Jan-1978
       to the date.

   INPUTS
       AmigaTime        - the number of seconds from 01-Jan-1978.

   RESULTS
       Date             - filled in with the date/time specified by
                          AmigaTime.

   NOTES

   SEE ALSO
       CheckDate(), Date2Amiga()

   BUGS
```

utility.library Page 4

```
utility.library/CallHookPkt                          utility.library/CallHookPkt

   NAME
       CallHookPkt -- Invoke a Hook function callback. (V36)

   SYNOPSIS
       return = CallHookPkt( hook, object, paramPkt )
       D0                    A0    A2      A1

       ULONG CallHookPkt( struct Hook *hook, VOID *object, VOID *paramPkt );

   FUNCTION
       Performs the callback standard defined by a Hook structure.
       This function is really very simple; it effectively performs
       a JMP to Hook->h_Entry.

       It is probably just as well to do this operation in an
       assembly language function linked in to your program, possibly
       from a compiler supplied library or a builtin function.

       It is anticipated that C programs will often call a 'varargs'
       variant of this function which will be named CallHook.  This
       function must be provided in a compiler specific library, but
       an example of use would be:

       returnval = CallHook( hook, dataobject, COMMAND_ID, param1, param2 );

       This function CallHook can be implemented in many C compilers
       like this:
       CallHook( hook, object, command, ... )
       struct Hook     *hook;
       VOID            *object;
       ULONG           command;
       {
               return ( CallHookPkt( hook, object, (VOID *) &command ) );
       }

   INPUTS
       Hook            - pointer to Hook structure as defined in
                         utility/hooks.h
       Object          - useful data structure in the particular context the
                         hook is being used for.
       ParamPkt        - pointer to a parameter packet (often built on the
                         stack); by convention this packet should start off
                         with a longword command code, and the remaining
                         data in the packet depends on that command.

   RESULT
       return          - The meaning of the value returned in D0 depends on
                         the context in which the Hook is being used.

   NOTES
       The functions called through this function should follow normal
       register conventions unless EXPLICITLY documented otherwise (and
       they have a good reason too).

   BUGS

   SEE ALSO
       utility/hooks.h
```

```
utility.library/CheckDate                              utility.library/CheckDate

   NAME
        CheckDate -- Checks ClockData struct for legal date. (V36)

   SYNOPSIS
        AmigaTime = CheckDate( Date )
        D0                     A0

        ULONG CheckDate( struct ClockData * );

   FUNCTION
        Determines if the Date is a legal date and returns the number
        of seconds to Date from 01-Jan-1978 if it is.

   INPUTS
        Date            - pointer to a ClockData structure.

   RESULTS
        AmigaTime       - 0 if Date invalid; otherwise, the number of
                          seconds to Date from 01-Jan-1978.

   NOTES

   BUGS
        The wday field of the ClockData structure is not checked.

   SEE ALSO
        Amiga2Date(), Date2Amiga()
```

```
utility.library/CloneTagItems                      utility.library/CloneTagItems

    NAME
        CloneTagItems -- Copies a TagItem list.  (V36)

    SYNOPSIS
        newTagList = CloneTagItems( tagList )
        D0                          A0

        struct TagItem *CloneTagItems( struct TagItem *tagList );

    FUNCTION
        Copies the essential contents of a tagItem list.  Internally,
        it uses AllocateTagItems() so that you can use FreeTagItems().

    INPUTS
        tagList         - TagItem list to clone.

    RESULT
        newTagList      - resultant copy.

    BUGS

    SEE ALSO
        AllocateTagItems(), FreeTagItems(), RefreshTagItemClones()
```

utility.library

Page 7

```
utility.library/Date2Amiga                          utility.library/Date2Amiga

   NAME
       Date2Amiga -- Calculate seconds from 01-Jan-1978.  (V36)

   SYNOPSIS
       AmigaTime = Date2Amiga( Date )
       D0                      A0

       ULONG Date2Amiga( struct ClockData * );

   FUNCTION
       Calculates the number of seconds from 01-Jan-1978 to the date
       specified in the ClockData structure.

   INPUTS
       Date              - pointer to a ClockData structure containing the
                           date of interest.

   RESULTS
       AmigaTime         - the number of seconds from 01-Jan-1978 to the
                           date specified in Date.

   NOTES
       This function does no sanity checking of the data in Date.

   SEE ALSO
       Amiga2Date(), CheckDate()

   BUGS
```

utility.library

Page 8

```
utility.library/FilterTagChanges             utility.library/FilterTagChanges

    NAME
        FilterTagChanges -- Eliminate TagItems which specify no change. (V36)

    SYNOPSIS
        FilterTagChanges( changeList, oldValues, apply)
                          A0          A1         D0

        void FilterTagChanges( struct TagItem *changeList,
                               struct TagItem *oldValues, LONG apply );

    FUNCTION
        Eliminate items from a "change list" that specify values already
        in effect in existing list.  Optionally update the existing list
        if the Boolean 'Apply' is true.

        The elimination is done by changing the ti_Tag field to TAG_IGNORE.
        So, this function may change the input tag_list(s).

    INPUTS
        changeList      - specification of new tag-value pairs.
        oldValues       - a list of existing tag item pairs.
        apply           - Boolean specification as to whether the values in
                          oldValues are to be updated to the values in
                          changeList.

    RESULT
        None.

    EXAMPLE
        Assume you have an attribute list for an object (oldValues)
        which looks like this:

                ATTR_Size,  "large",
                ATTR_Color, "orange",
                ATTR_Shape, "square",

        If you receive a new TagList containing some changes (changeList),
        which looks like this:

                ATTR_Size,  "large",
                ATTR_Shape, "triangle"

        If you call FilterTagChanges(), changeList will be modified to
        contain only those attributes which are different from the
        oldValues.  All other tagitems will have their tag-values set to
        TAG_IGNORE.  The resulting changeList will become:

                TAG_IGNORE, "large",
                ATTR_Shape, "triangle"

        If apply was set to TRUE, oldValues would be:

                ATTR_Size,  "large"
                ATTR_Color, "orange"
                ATTR_Shape, "triangle"

    BUGS

    SEE ALSO
```

```
utility.library/FilterTagItems                     utility.library/FilterTagItems

    NAME
        FilterTagItems - Remove selected items from a TagItem list. (V36)

    SYNOPSIS
        nvalid = FilterTagItems(tagList, tagArray, logic)
        D0                      A0       A1        D0

        ULONG FilterTagItems(struct TagItem *tagList, Tag *tagArray,
                             LONG logic);

    FUNCTION
        Removes TagItems from a TagItem list (by changing ti_Tag to
        TAG_IGNORE) depending on whether its ti_Tag value is
        found in an array of TagValues.

        If the 'logic' parameter is TAGFILTER_AND, then all items
        not appearing in the list are excluded.

        If 'logic' is TAGFILTER_NOT, then items not found in the
        array are preserved, and the ones in the array are cast out.

    INPUTS
        tagList         - input list of tag items which is to be filtered
                          by having selected items changed to TAG_IGNORE.
        tagArray        - an array of Tag values, terminated by TAG_END,
                          as specified in the notes on TagInArray().
        logic           - specification whether items in TagArray are to
                          be included or excluded in the filtered result.

    RESULT
        nvalid          - number of valid items left in resulting filtered
                          list.

    BUGS

    SEE ALSO
        TagInArray()
```

```
utility.library/FindTagItem                           utility.library/FindTagItem

    NAME
        FindTagItem -- Scans TagItem list for a Tag. (V36)

    SYNOPSIS
        tag = FindTagItem( tagVal, tagList)
        D0                 D0      A0

        struct TagItem *FindTagItem( Tag tagVal, struct TagItem *tagList );

    FUNCTION
        Scans a TagItem "List", which is in fact a chain of arrays
        of TagItem structures as defined in utility/tagitem.h.
        Returns a pointer to the FIRST item with ti_Tag matching the
        'TagVal' parameter.

    INPUTS
        tagVal          - Tag value to search for.
        tagList         - beginning of TagItem list to scan.

    RESULT
        Returns a pointer to the item with ti_Tag matching 'TagVal'.
        Returns NULL if there is no match or if TagList is NULL.

    BUGS

    SEE ALSO
        utility/tagitem.h, GetTagData(), PackBoolTags(), NextTagItem()
```

```
utility.library/FreeTagItems                      utility.library/FreeTagItems

    NAME
        FreeTagItems --  Frees allocated TagItem lists. (V36)

    SYNOPSIS
        FreeTagItems( tagList )
                      A0

        void FreeTagItems( struct TagItem *tagList );

    FUNCTION
        Frees the memory of a TagItem list allocated either by
        AllocateTagItems() or CloneTagItems().

    INPUTS
        TagList         - list to free.  Must be created by functions
                          specified. A value of NULL for 'tagList' is safe.

    RESULT
        None.

    BUGS

    SEE ALSO
        AllocateTagItems(), CloneTagItems()
```

```
utility.library/GetTagData                              utility.library/GetTagData

    NAME
        GetTagData -- Obtain data corresponding to Tag. (V36)

    SYNOPSIS
        value = GetTagData(tagVal, default, tagList)
        D0                 D0      D1       A0

        ULONG GetTagData(Tag TagVal, ULONG Default, struct TagItem *TagList)

    FUNCTION
        Searches a TagItem list for a matching Tag value, and returns the
        corresponding ti_Data value for the TagItem found.  If none
        found, will return the value passed it as 'default'.

    INPUTS
        tagVal          - Tag value to search for.
        default         - value to be returned if tagVal is not found.
        tagList         - the TagItem list to search.

    RESULT
        value           - The ti_Data value for first matching TagItem, or
                          'default' if a ti_Tag matching 'Tag' is not found.

    BUGS

    SEE ALSO
        utility/tagitem.h, FindTagItem(), PackBoolTags(), NextTagItem()
```

```
utility.library/MapTags                                  utility.library/MapTags

    NAME
        MapTags -- Convert ti_Tag values in a list via map pairing. (V36)

    SYNOPSIS
        MapTags(tagList, mapList, includeMiss)
                A0       A1       D0

        void MapTags(struct TagItem *tagList, struct TagItem mapList,
                     LONG includeMiss);

    FUNCTION
        Apply a "mapping list" mapList to tagList:

        If the ti_Tag field of an item in tagList appears as ti_Tag in some
        item in mapList, overwrite ti_Tag with the corresponding ti_Data
        from the map list.

        If a tag in tagList does not appear in the mapList, you can choose
        to have it removed by changing it to TAG_IGNORE. Do this by setting
        includeMiss to FALSE.

        If you want to have items which do not appear in the mapList
        survive the mapping as-is, set includeMiss to 1.

        This is central to gadget interconnections where you want
        to convert the tag values from one space (the sender) to
        another (the receiver).

    INPUTS
        tagList         - Input list of tag items which is to be mapped
                          to Tag values as specified in mapList.
        mapList         - a "mapping list" tagItem list which pairs Tag
                          values expected to appear in tagList with new
                          values to be substituted in the ti_Tag fields of
                          tagList.  May be NULL, which means that all items
                          in tagList will be eliminated.
        includeMiss     - 0 to remove tags from tagList not in mapList,
                          1 to remove

    RESULT
        None.

    EXAMPLE
        /* Consider this source list: */
            struct TagItem list[] = {
                { MY_SIZE,      71 },
                { MY_WEIGHT,    200 },
                { TAG_END,      } };

        /* And the mapping list: */
            struct TagItem map[] = {
                { MY_SIZE,      HIS_TALL },
                { TAG_END,      } };

        /* Then after MapTags( list, map, 0 ), 'list' will become: */
                { HIS_TALL, 71 },
                { TAG_IGNORE, },
                { TAG_END, }

        /* Then after MapTags( list, map, 1 ), 'list' will become: */
                { HIS_TALL, 71 },
                { MY_WEIGHT, 200 },
                { TAG_END, }
```

```
    NOTES
        The procedure will change the values of the input tag list
        tagList (but not mapList).

        You can "filter" a list by passing includeMiss as 0, and having the
        data items in the map list equal the corresponding tags.

        You can perform the inverse filter ("everything but") by passing
        includeMiss equal to 1, and creating a map item for every tag you
        want to filter out, pairing it with a mapped data value of
        TAG_IGNORE.

        For safety and "order independence" of tag item arrays, if you
        attempt to map some tag to the value TAG_END, the value TAG_IGNORE
        will be substituted instead.

    BUGS

    SEE ALSO
```

utility.library Page 15

```
utility.library/NextTagItem                          utility.library/NextTagItem

    NAME
        NextTagItem -- Iterate TagItem lists. (V36)

    SYNOPSIS
        next_tag = NextTagItem( tagItemPtr )
        D0                      A0

        struct TagItem *NextTagItem( struct TagItem **tagItemPtr );

    FUNCTION
        Iterates through a (chained) array of TagItem structures,
        skipping and chaining as dictated by system tags.  TAG_SKIP
        will cause it to skip the entry and the next, TAG_IGNORE ignores
        that single entry, and TAG_MORE has a pointer to another array
        of tags (and terminates the current array!)  TAG_DONE also
        terminates the current array.  Each call returns either the next
        tagitem you should examine, or NULL at the end.

    INPUTS
        tagItemPtr      - doubly-indirect reference to a TagItem structure.
                          The pointer will be changed to keep track of the
                          iteration.

    RESULT
        next_tag        - Each TagItem in the array or chain of arrays that
                          should be processed according to system Tag values
                          (in utility/tagitem.h) is returned in turn with
                          successive calls.

    EXAMPLE
        Iterate( struct TagItem *tags );
        {
                struct TagItem *tstate;
                struct TagItem *tag;

                tstate = tags;
                while ( tag = NextTagItem( &tstate ) )
                {
                        switch ( tag->ti_Tag )
                        {
                        case TAG1:
                            ...
                            break;
                        case TAG2:
                            ...
                            break;
                        ...
                        }
                }
        }

    NOTES
        Do NOT use the value of *tagItemPtr, but rather use the pointer
        returned by NextTagItem().


    BUGS

    SEE ALSO
        utility/tagitem.h, GetTagData(), PackBoolTags(), FindTagItem()
```

utility.library Page 16

```
utility.library/PackBoolTags                        utility.library/PackBoolTags

    NAME
        PackBoolTags -- Builds a "Flag" word from a TagList. (V36)

    SYNOPSIS
        boolflags = PackBoolTags( initialFlags, tagList, boolMap )
        D0                        D0            A0       A1

        ULONG PackBoolTags( ULONG initialFlags, struct TagItem *tagList,
                            struct TagItem *boolMap );

    FUNCTION
        Picks out the Boolean TagItems in a TagItem list and converts
        them into bit-flag representations according to a correspondence
        defined by the TagItem list 'BoolMap.'

        A Boolean TagItem is one where only the logical value of
        the ti_Data is relevant.  If this field is 0, the value is
        FALSE, otherwise TRUE.


    INPUTS
        initialFlags    - a starting set of bit-flags which will be changed
                          by the processing of TRUE and FALSE Boolean tags
                          in tagList.
        tagList         - a TagItem list which may contain several TagItems
                          defined to be "Boolean" by their presence in
                          boolMap.  The logical value of ti_Data determines
                          whether a TagItem causes the bit-flag value related
                          by boolMap to set or cleared in the returned flag
                          longword.
        boolMap         - a TagItem list defining the Boolean Tags to be
                          recognized, and the bit (or bits) in the returned
                          longword that are to be set or cleared when a
                          Boolean Tag is found to be TRUE or FALSE in
                          tagList.

    RESULT
        boolflags       - the accumulated longword of bit-flags, starting
                          with InitialFlags and modified by each Boolean
                          TagItem encountered.

    EXAMPLE

        /* define some nice user tag values ... */
        enum mytags { tag1 = TAG_USER+1, tag2, tag3, tag4, tag5 };

        /* this TagItem list defines the correspondence between Boolean tags
         * and bit-flag values.
         */
        struct TagItem  boolmap[] = {
            { tag1,  0x0001 },
            { tag2,  0x0002 },
            { tag3,  0x0004 },
            { tag4,  0x0008 },
            { TAG_DONE }
        };

        /* You are probably passed these by some client, and you want
         * to "collapse" the Boolean content into a single longword.
         */

        struct TagItem  boolexample[] = {
            { tag1,  TRUE },
            { tag2,  FALSE },
        { tag5, Irrelevant },
        { tag3,  TRUE },
        { TAG_DONE }
    };

    /* Perhaps 'boolflags' already has a current value of 0x800002. */
    boolflags = PackBoolTags( boolflags, boolexample, boolmap );

    /* The resulting new value of 'boolflags' will be 0x80005. /*

BUGS
    There are some undefined cases if there is duplication of
    a given Tag in either list.  It is probably safe to say that
    the *last* of identical Tags in TagList will hold sway.

SEE ALSO
    utility/tagitem.h, GetTagData(), FindTagItem(), NextTagItem()
```

```
utility.library/RefreshTagItemClones     utility.library/RefreshTagItemClones

   NAME
       RefreshTagItemClones -- Rejuvenates a clone from the original. (V36)

   SYNOPSIS
       RefreshTagItemClones( cloneTagItems, originalTagItems )
                             A0             A1

       void RefreshTagItemClones( struct TagItem *cloneTagItems,
                                  struct TagItem *originalTagItems );

   FUNCTION
       If (and only if) the tag items 'cloneTagItems' were created
       from 'originalTagItems' by CloneTagItems(), and if originalTagItems
       has not been changed in any way, you can reset the clone list
       to its original state by using this function.

   INPUTS
       CloneTagItems   - return value from CloneTagItems(originalTagItems).
       OriginalTagItems- a tag list that hasn't changed.

   RESULT
       None.

   EXAMPLE

   BUGS

   SEE ALSO
       CloneTagItems(), AllocateTagItems(), FreeTagItems()
```

utility.library

Page 19

```
utility.library/SDivMod32                                  utility.library/SDivMod32

   NAME
        SDivMod32 -- Signed 32 by 32 bit division and modulus. (V36)

   SYNOPSIS
        Quotient:Remainder = SDivMod32( Dividend, Divisor )
        D0       D1                     D0        D1

        LONG SDivMod32( LONG, LONG );

   FUNCTION
        Divides the signed 32 bit dividend by the signed 32 bit divisor
           and returns a signed 32 bit quotient and remainder.

   INPUTS
        Dividend        - signed 32 bit dividend.
        Divisor         - signed 32 bit divisor.

   RESULTS
        Quotient        - signed 32 quotient of the division.
        Remainder       - signed 32 remainder of the division.

   NOTES

   SEE ALSO
        SMult32(), UDivMod32(), UMult32()

   BUGS
```

utility.library

Page 20

```
utility.library/SMult32                                      utility.library/SMult32

   NAME
        SMult32 -- Signed 32 by 32 bit multiply with 32 bit result. (V36)

   SYNOPSIS
        Result = SMult32( Arg1, Arg2 )
        D0                D0    D1

        LONG SMult32( LONG, LONG );

   FUNCTION
        Returns the signed 32 bit result of multiplying Arg1 by Arg2.

   INPUTS
        Arg1, Arg2      - signed multiplicands.

   RESULTS
        Result          - the signed 32 bit result of multiplying
                          Arg1 by Arg2.

   NOTES

   SEE ALSO
        SDivMod32(), UDivMod32(), UMult32()

   BUGS
```

```
utility.library/Stricmp                               utility.library/Stricmp

   NAME
       Stricmp --  Case-insensitive string compare. (V37)

   SYNOPSIS
       res = Stricmp(string1, string2)
       D0            A0       A1

       LONG Stricmp(char *, char *);

   FUNCTION
       Stricmp compares two strings, ignoring case.  It handles all
       internationalization issues.  If the strings have different lengths,
       the shorter is treated as if it were extended with zeros.

   INPUTS
       string1, string2 - strings to be compared

   RESULT
       res - negative if string1 is below string2, 0 if they're the same, and
             positive if string1 is above string2.

   NOTES
       Commodore is planning a localization library which will take care
       of most details pertaining to system integration into different
       cultures, locales, and territories.

       This function will automatically be replaced by a localized version
       whenever the locale.library is loaded in memory. As such, the
       collating order may change depending on the locale currently
       defined by the user. Take this fact into consideration when using
       this function, and do not rely on obtaining specific collating
       sequences.

   BUGS

   SEE ALSO
       Strnicmp()
```

```
utility.library/Strnicmp                             utility.library/Strnicmp

   NAME
       Strnicmp--  Case-insensitive string compare, length-limited. (V37)

   SYNOPSIS
       res = Strnicmp(string1, string2, length)
       D0             A0       A1       D0

       LONG Strnicmp(char *, char *, LONG length);

   FUNCTION

       Strnicmp compares two strings, ignoring case.  It handles all
       internationalization issues.  If the strings have different lengths,
       the shorter is treated as if it were extended with zeros.  It never
       compares more than <length> characters.

   INPUTS
       string1, string2 - strings to be compared
       length           - maximum number of characters to examine

   RESULT
       res - negative if string1 is below string2, 0 if they're the same, and
             positive if string1 is above string2.

   NOTES
       Commodore is planning a localization library which will take care
       of most details pertaining to system integration into different
       cultures, locales, and territories.

       This function will automatically be replaced by a localized version
       whenever the locale.library is loaded in memory. As such, the
       collating order may change depending on the locale currently
       defined by the user. Take this fact into consideration when using
       this function, and do not rely on obtaining specific collating
       sequences.

   BUGS

   SEE ALSO
       Stricmp()
```

utility.library

```
utility.library/TagInArray                          utility.library/TagInArray

    NAME
        TagInArray  -- Check if a Tag value appears in a Tag array. (V36)

    SYNOPSIS
        BOOL TagInArray( tag, tagArray )
        D0               D0   A0

        BOOL TagInArray( Tag tag, Tag *tagArray);

    FUNCTION
        Perform a quick scan to see if a tag value appears in
        an array terminated with TAG_END.  Returns TRUE if
        the value is found.

        The 'tagArray' must be terminated by TAG_END.  It should
        NOT contain other system tag values, such as TAG_MORE
        or TAG_IGNORE.  Note that this is an array of Tag values, NOT
        an array of TagItems.

        This is sort of a "one shot" version of FilterTagItems().

    INPUTS
        tag            - Tag value to search array for.
        tagArray       - a simple array terminated by TAG_END.

    RESULT
        Boolean success of search.

    BUGS

    SEE ALSO
        FilterTagItems()
```

utility.library

```
utility.library/ToLower                                utility.library/ToLower

    NAME
        ToLower - Convert a character to lowercase. (V37)

    SYNOPSIS
        char = ToLower(char)
        D0             D0

        char ToLower(char);

    FUNCTION
        Converts a character to lowercase, handling international character
        sets.

    INPUTS
        char - character to be converted.

    RESULT
        char - lowercase version of input character.

    NOTES
        Commodore is planning a localization library which will take care
        of most details pertaining to system integration into different
        cultures, locales, and territories.

        This function will automatically be replaced by a localized version
        whenever the locale.library is loaded in memory. As such, the
        resulting converted character may change depending on the locale
        currently defined by the user. Take this fact into consideration when
        using this function, and do not rely on obtaining specific
        conversions.

    BUGS

    SEE ALSO
```

```
utility.library/ToUpper                             utility.library/ToUpper

    NAME
        ToUpper - Convert a character to uppercase. (V37)

    SYNOPSIS
        char = ToUpper(char)
        D0             D0

        char ToUpper(char);

    FUNCTION
        Converts a character to uppercase, handling international character
        sets.

    INPUTS
        char - character to be converted.

    RESULT
        char - uppercase version of input character.

    NOTES
        Commodore is planning a localization library which will take care
        of most details pertaining to system integration into different
        cultures, locales, and territories.

        This function will automatically be replaced by a localized version
        whenever the locale.library is loaded in memory. As such, the
        resulting converted character may change depending on the locale
        currently defined by the user. Take this fact into consideration when
        using this function, and do not rely on obtaining specific
        conversions.

    BUGS

    SEE ALSO
```

```
utility.library/UDivMod32                         utility.library/UDivMod32

    NAME
        UDivMod32 -- Unsigned 32 by 32 bit division and modulus. (V36)

    SYNOPSIS
        Quotient:Remainder = UDivMod32( Dividend, Divisor )
        D0       D1                     D0        D1

        ULONG UDivMod32( ULONG, ULONG );

    FUNCTION
        Divides the unsigned 32 bit dividend by the unsigned 32 bit divisor
        and returns a unsigned 32 bit quotient and remainder.

    INPUTS
        Dividend        - unsigned 32 bit dividend.
        Divisor         - unsigned 32 bit divisor.

    RESULTS
        Quotient        - unsigned 32 quotient of the division.
        Remainder       - unsigned 32 remainder of the division.

    NOTES

    SEE ALSO
        SDivMod32(), SMult32(), UMult32()

    BUGS
```

```
utility.library/UMult32                              utility.library/UMult32

   NAME
        UMult32 -- Unsigned 32 by 32 bit multiply with 32 bit result. (V36)

   SYNOPSIS
        Result = UMult32( Arg1, Arg2 )
        D0                D0    D1

        ULONG UMult32( ULONG, ULONG );

   FUNCTION
        Returns the unsigned 32 bit result of multiplying Arg1 by Arg2.

   INPUTS
        Arg1, Arg2              - unsigned multiplicands.

   RESULTS
        Result                  - the unsigned 32 bit result of
                                  multiplying Arg1 by Arg2.

   NOTES

   SEE ALSO
        SDivMod32(), SMult32(), UDivMod32()

   BUGS
```

TABLE OF CONTENTS

workbench.library/AddAppIconA
workbench.library/AddAppMenuItemA
workbench.library/AddAppWindowA
workbench.library/RemoveAppIcon
workbench.library/RemoveAppMenuItem
workbench.library/RemoveAppWindow

```
workbench.library/AddAppIconA                   workbench.library/AddAppIconA

   NAME
        AddAppIcon - add an icon to workbench's list of appicons.        (V36)

   SYNOPSIS
        AppIcon = AddAppIconA(id, userdata, text, msgport,
          D0                  D0      D1      A0    A1

                                      lock, diskobj, taglist)
                                       A2      A3      A4

        struct AppIcon *AddAppIconA(ULONG, ULONG, char *,
                struct MsgPort *, struct FileLock *, struct DiskObject *,
                struct TagItem *);

        Alternate, varargs version:
        struct AppIcon *AddAppIcon(ULONG, ULONG, char *,
                        struct MsgPort *, struct FileLock *,
                        struct DiskObject *,
                        tag1, data1,
                        tag2, data2,
                        ...
                        TAG_END );

   FUNCTION
        Attempt to add an icon to workbench's list of appicons.  If
        successful, the icon is displayed on the workbench (the same
        place disk icons are displayed).
        This call is provided to allow applications to be notified when
        a graphical object (non neccessarely associated with a file)
        gets 'manipulated'. (explained later).
        The notification consists of an AppMessage (found in workbench.h/i)
        of type 'MTYPE_APPICON' arriving at the message port you specified.
        The types of 'manipulation' that can occur are:
        1. Double-clicking on the icon.  am_NumArgs will be zero and
           am_ArgList will be NULL.
        2. Dropping an icon or icons on your appicon.  am_NumArgs will
          be the number of icons droppped on your app icon plus one.
          am_ArgList will be an array of ptrs to WBArg structures.
          Refer to the 'WBStartup Message' section of the RKM for more info.
        3. Dropping your appicon on another icon.  NOT SUPPORTED.

   INPUTS
        id - this variable is strictly for your own use and is ignored by
             workbench.  Typical uses in C are in switch and case statements,
             and in assembly language table lookup.
        userdata - this variable is strictly for your own use and is ignored
                   by workbench.
        text - name of icon (char *)
        lock - NULL    (Currently unused)
        msgport - pointer to message port workbench will use to send you an
                  AppMessage message of type 'MTYPE_APPICON' when your icon
                  gets 'manipulted' (explained above).
        diskobj - pointer to a DiskObject structure filled in as follows:
            do_Magic - NULL
            do_Version - NULL
            do_Gadget - a gadget structure filled in as follows:
                NextGadget - NULL
                LeftEdge - NULL
                TopEdge - NULL
                Width - width of icon hit-box
                Height - height of icon hit-box
                Flags - NULL or GADGHIMAGE
                Activation - NULL
                GadgetType - NULL
            GadgetRender - pointer to Image structure filled in as follows:
                LeftEdge - NULL
                TopEdge - NULL
                Width - width of image (must be <= Width of hit box)
                Height - height of image (must be <= Height of hit box)
                Depth - # of bit-planes in image
                ImageData - pointer to actual word aligned bits (CHIP MEM)
                PlanePick - Plane mask ((1 << depth) - 1)
                PlaneOnOff - 0
                NextImage - NULL
            SelectRender - pointer to alternate Image struct or NULL
            GadgetText - NULL
            MutualExclude - NULL
            SpecialInfo - NULL
            GadgetID - NULL
            UserData - NULL
        do_Type - NULL
        do_DefaultTool - NULL
        do_ToolTypes - NULL
        do_CurrentX - NO_ICON_POSITION (recommended)
        do_CurrentY - NO_ICON_POSITION (recommended)
        do_DrawerData - NULL
        do_ToolWindow - NULL
        do_StackSize - NULL

    (an easy way to create one of these (a DiskObject) is to create an icon
    with the V2.0 icon editor and save it out.  Your application can then
    call GetDiskObject on it and pass that to AddAppIcon.)

    taglist - ptr to a list of tag items.  Must be NULL for V2.0.

RESULTS
    AppIcon - a pointer to an appicon structure which you pass to
              RemoveAppIcon when you want to remove the icon
              from workbench's list of appicons.  NULL
              if workbench was unable to add your icon; typically
              happens when workbench is not running or under low
              memory conditions.

EXAMPLE
    You could design a print-spooler icon and add it to the workbench.
    Any file dropped on the print spooler would be printed.  If the
    user double-clicked (opended) your printer-spooler icon, you could
    open a window showing the status of the print spool, allow changes
    to print priorities, allow deletions, etc.  If you registered this
    window as an 'appwindow' (explained in workbench.library AddAppWindow)
    files could also be dropped in the window and added to the spool.

SEE ALSO
    RemoveAppIcon()

BUGS
    Currently Info cannot be obtained on appicons.
```

```
workbench.library/AddAppMenuItemA           workbench.library/AddAppMenuItemA

   NAME
        AddAppMenuItem - add a menuitem to workbench's list             (V36)
                         of appmenuitems.

   SYNOPSIS
        AppMenuItem = AddAppMenuItemA(id, userdata, text, msgport, taglist)
        D0                            D0     D1       A0     A1       A2

        struct AppMenuItem *AddAppMenuItemA(ULONG, ULONG, char *,
                                            struct MsgPort *,
                                            struct TagItem *);

        Alternate, varargs version:
        struct AppMenuItem *AddAppMenuItem(ULONG, ULONG, char *,
                                           struct MsgPort *,
                                           tag1, data1,
                                           tag2, data2,
                                           ...
                                           TAG_END );

   FUNCTION
        Attempt to add the text as a menuitem to workbench's list
        of appmenuitems (the 'Tools' menu strip).

   INPUTS
        id - this variable is strictly for your own use and is ignored by
             workbench.  Typical uses in C are in switch and case statements,
             and in assembly language table lookup.
        userdata - this variable is strictly for your own use and is ignored
                   by workbench.
        text - text for the menuitem (char *)
        msgport - pointer to message port workbench will use to send you an
                  AppMessage message of type 'MTYPE_APPMENUITEM' when your
                  menuitem gets selected.
        taglist - ptr to a list of tag items.  Must be NULL for V2.0.

   RESULTS
        AppMenuItem - a pointer to an appmenuitem structure which you pass to
                      RemoveAppMenuItem when you want to remove the menuitem
                      from workbench's list of appmenuitems.  NULL if
                      workbench was unable to add your menuitem; typically
                      happens when workbench is not running or under low
                      memory conditions.

   SEE ALSO
        RemoveAppMenuItem()

   BUGS
        Currently does not limit the system to 63 menu items...
        Any menu items after the 63rd will not be selectable.
```

```
workbench.library/AddAppWindowA                    workbench.library/AddAppWindowA

   NAME
        AddAppWindow - add a window to workbench's list of appwindows.  (V36)

   SYNOPSIS
        AppWindow = AddAppWindowA(id, userdata, window, msgport, taglist)
        D0                        D0     D1       A0       A1       A2

        struct AppWindow *AddAppWindowA(ULONG, ULONG, struct Window *,
                                        struct MsgPort *, struct TagItem *);

        Alternate, varargs version:
        struct AppWindow *AddAppWindow(ULONG, ULONG, struct Window *,
                                        struct MsgPort *
                                        tag1, data1,
                                        tag2, data2,
                                        ...
                                        TAG_END );

   FUNCTION
        Attempt to add the window to workbench's list of appwindows.
        Normally non-workbench windows (those not opened by workbench)
        cannot have icons dropped in them.  This call is provided to
        allow applications to be notified when an icon or icons get
        dropped inside a window that they have registered with workbench.
        The notification consists of an AppMessage (found in workbench.h/i)
        of type 'MTYPE_APPWINDOW' arriving at the message port you specified.
        What you do with the list of icons (pointed to by am_ArgList) is
        up to you, but generally you would want to call GetDiskObjectNew on
        them.

   INPUTS
        id - this variable is strictly for your own use and is ignored by
             workbench.  Typical uses in C are in switch and case statements,
             and in assembly language table lookup.
        userdata - this variable is strictly for your own use and is ignored
                   by workbench.
        window - pointer to window to add.
        msgport - pointer to message port workbench will use to send you an
                  AppMessage message of type 'MTYPE_APPWINDOW' when your
                  window gets an icon or icons dropped in it.
        taglist - ptr to a list of tag items.  Must be NULL for V2.0.

   RESULTS
        AppWindow - a pointer to an appwindow structure which you pass to
                    RemoveAppWindow when you want to remove the window
                    from workbench's list of appwindows.  NULL
                    if workbench was unable to add your window; typically
                    happens when workbench is not running or under low
                    memory conditions.

   SEE ALSO
        RemoveAppWindow()

   NOTES
       The V2.0 icon editor is an example of an app window.  Note that app
       window applications generally want to call GetDiskObjectNew
       (as opposed to GetDiskObject) to get the disk object for the icon
       dropped in the window.

   BUGS
        None
```

```
workbench.library/RemoveAppIcon                    workbench.library/RemoveAppIcon

   NAME
        RemoveAppIcon - remove an icon from workbench's list            (V36)
                        of appicons.

   SYNOPSIS
        error = RemoveAppIcon(AppIcon)
         D0                    A0
        BOOL RemoveAppIcon(struct AppIcon *);

   FUNCTION
        Attempt to remove an appicon from workbench's list of appicons.

   INPUTS
        AppIcon - pointer to an AppIcon structure returned by AddAppIcon.

   RESULTS
        error - Currently always TRUE...

   NOTES
        As with anything that deals with async operation, you will need to
        do a final check for messages on your App message port for messages
        that may have come in between the last time you checked and the
        call to removed the App.

   SEE ALSO
        AddAppIcon()

   BUGS
        None
```

workbench.library Page 7

```
workbench.library/RemoveAppMenuItem        workbench.library/RemoveAppMenuItem

   NAME
       RemoveAppMenuItem - remove a menuitem from workbench's list      (V36)
                           of appmenuitems.

   SYNOPSIS
       error = RemoveAppMenuItem(AppMenuItem)
        D0                         A0
       BOOL RemoveAppMenuItem(struct AppMenuItem *);

   FUNCTION
       Attempt to remove an appmenuitem from workbench's list
       of appmenuitems.

   INPUTS
       AppMenuItem - pointer to an AppMenuItem structure returned by
                     AddAppMenuItem.

   RESULTS
       error - Currently always TRUE...

   NOTES
       As with anything that deals with async operation, you will need to
       do a final check for messages on your App message port for messages
       that may have come in between the last time you checked and the
       call to removed the App.

   SEE ALSO
       AddAppMenuItem()

   BUGS
       None
```

---

workbench.library Page 8

```
workbench.library/RemoveAppWindow            workbench.library/RemoveAppWindow

   NAME
       RemoveAppWindow - remove a window from workbench's list          (V36)
                         of appwindows.

   SYNOPSIS
       error = RemoveAppWindow(AppWindow)
        D0                       A0
       BOOL RemoveAppWindow(struct AppWindow *);

   FUNCTION
       Attempt to remove an appwindow from workbench's list of appwindows.

   INPUTS
       AppWindow - pointer to an AppWindow structure returned by
                   AddAppWindow.

   RESULTS
       error - Currently always TRUE...

   NOTES
       As with anything that deals with async operation, you will need to
       do a final check for messages on your App message port for messages
       that may have come in between the last time you checked and the
       call to removed the App.

   SEE ALSO
       AddAppWindow()

   BUGS
       None
```

Device Autodocs

This section contains summaries for the device calls that are built into the Amiga's operating system software. These summaries have been automatically extracted from the system source code and are called Autodocs.

Devices are based on the library concept described in the last section. But unlike libraries, devices use a standard command interface. Once you have learned how to use one Amiga device, it's easy to learn the others.

An Amiga device is a hardware-independent software module for interacting with a physical I/O device (such as a disk drive or serial port). Devices typically have their own independent tasks that can perform asynchronous operations even while the task that called them is busy doing something else.

For a more detailed introduction to the Amiga device concept refer to the *Amiga ROM Kernel Reference Manual: Devices*. Only a brief introduction is given here. The table below shows the names of all the devices that are currently part of the Amiga system software.

Amiga Devices

audio.device
clipboard.device
console.device
gameport.device
input.device
keyboard.device
narrator.device
parallel.device
printer.device
serial.device
timer.device
trackdisk.device

Devices are more complex to use than libraries. Opening a device requires:

- ☐ A message port structure (`struct MsgPort`). This structure is used for inter-task communication. Ports may be created with the amiga.lib/CreatePort() function. See the section on Linker Libraries for more information on CreatePort().

- ☐ An I/O Request structure (`struct IORequest` or one of its extended forms). This special structure, plus any extensions, is your sole source of communication with the device. Commands and data (or data pointers) are placed in this structure and sent off to the device.

  The format of the basic IORequest structure is defined in the exec/io.h listing (see the section on Include Files). Some devices use a special extended version of the IORequest structure. If so, the extended IORequest is defined in the include file for the particular device that uses it. An IORequest can be created with the amiga.lib/CreateExtIO() function. See the Linker Libraries section for more information on CreateExtIO().

- ☐ The name of the device for the Exec OpenDevice() call. The names of Amiga devices are shown in the table on page 461.

Opening the device initializes the IORequest for use. The request will always be associated with the device that initialized it so you cannot use it with other devices. Once the device is opened, you can send commands to it. Commands are placed in the `io_Command` field of the IORequest, then the request is sent to the device. There are two primary options for sending the request:

DoIO() - An Exec call that does the I/O and returns only after it has finished (this is synchronous I/O). This is the easiest option to use.

SendIO() - An Exec call that starts the I/O, but returns immediately (this is asynchronous I/O). The device will complete its job while your calling task continues to run. Before reusing the IORequest, you must wait for the I/O to finish. You can send multiple requests to a device and the requests will queue up. However, you must use separate IORequest structures for multiple requests.

When you have finished using a device, a call to CloseDevice() completes the transaction. For those programs using asynchronous I/O, any outstanding requests must have already been completed. This can be done by a WaitIO(), or by forcing termination with an AbortIO()/WaitIO() pair.

```
/*
* A complete example of using the trackdisk.device.
* This moves the heads from track 0 to 79 and back.
*
*/
#include "exec/types.h"
#include "devices/trackdisk.h"
#include "dos/dos.h"

struct MsgPort *trackport=NULL;   /* Storage for pointers */
struct IOExtTD *trackIO=NULL;   /* An extended version of struct IORequest */
short         openerror=NULL;     /* flag */

void cleanexit(returncode)
int returncode;
{
       printf("openerror =%ld\n"  ,openerror);
       printf("trackIO   =$%lx\n",trackIO);
       printf("trackport =$%lx\n",trackport);
       printf("io_Error  =%ld\n"  ,trackIO->iotd_Req.io_Error);

       if(!openerror) CloseDevice(trackIO);
       if(trackIO)    DeleteExtIO(trackIO);
       if(trackport)  DeletePort(trackport);

       exit(returncode);
}


void main()
{
       trackport=(struct MsgPort *)CreatePort(0L,0L);
       if(!trackport)
              cleanexit(RETURN_FAIL);
       trackIO=(struct IOExtTD *)
           CreateExtIO(trackport,(long)sizeof(struct IOExtTD));
       if(!trackIO)
              cleanexit(RETURN_FAIL+1);
       if(openerror=OpenDevice("trackdisk.device",0L,trackIO,0L))
              cleanexit(RETURN_FAIL+2);

       trackIO->iotd_Req.io_Command=TD_SEEK;      /* trackdisk command */

       trackIO->iotd_Req.io_Offset =0L;        /* outer track */
       printf("Track 0\n");
       DoIO(trackIO);
       trackIO->iotd_Req.io_Offset =79*11*2*512L;     /* inner track */
       printf("Track 79\n");
       DoIO(trackIO);
       trackIO->iotd_Req.io_Offset =0L;        /* outer track */
       printf("Track 0\n");
       DoIO(trackIO);
       trackIO->iotd_Req.io_Offset =79*11*2*512L;     /* inner track */
       printf("Track 79\n");
       DoIO(trackIO);

       cleanexit(RETURN_OK);
}
```

audio.device

Page 1

TABLE OF CONTENTS

audio.device/AbortIO
audio.device/BeginIO
audio.device/CloseDevice
audio.device/ADCMD_ALLOCATE
audio.device/ADCMD_FINISH
audio.device/ADCMD_FREE
audio.device/ADCMD_LOCK
audio.device/ADCMD_PERVOL
audio.device/ADCMD_SETPREC
audio.device/ADCMD_WAITCYCLE
audio.device/CMD_CLEAR
audio.device/CMD_FLUSH
audio.device/CMD_READ
audio.device/CMD_RESET
audio.device/CMD_START
audio.device/CMD_STOP
audio.device/CMD_UPDATE
audio.device/CMD_WRITE
audio.device/Expunge
audio.device/OpenDevice

audio.device

Page 2

```
audio.device/AbortIO                                          audio.device/AbortIO

   NAME
       AbortIO - abort a device command

   SYNOPSIS
       AbortIO(iORequest);
                   A1

   FUNCTION
       AbortIO tries to abort a device command.  It is allowed to be
       unsuccessful.  If the Abort is successful, the io_Error field of the
       iORequest contains an indication that IO was aborted.

   INPUTS
       iORequest -- pointer to the I/O Request for the command to abort
```

```
audio.device/BeginIO                                    audio.device/BeginIO

   NAME
       BeginIO - dispatch a device command

   SYNOPSIS
       BeginIO(iORequest);
                  A1

   FUNCTION
       BeginIO has the responsibility of dispatching all device commands.
       Immediate commands are always called directly, and all other commands
       are queued to make them single threaded.

   INPUTS
       iORequest -- pointer to the I/O Request for this command
```

```
audio.device/CloseDevice                            audio.device/CloseDevice

   NAME
       CloseDevice - terminate access to the audio device

   SYNOPSIS
       CloseDevice(iORequest);
                      A1

   FUNCTION
       The CloseDevice routine notifies the audio device that it will no
       longer be used.  It takes an I/O audio request block (IOAudio) and
       clears the device pointer (io_Device).  If there are any channels
       allocated with the same allocation key (ioa_AllocKey), CloseDevice
       frees (ADCMD_FREE) them. CloseDevice decrements the open count, and if
       it falls to zero and an expunge (Expunge) is pending, the device is
       expunged.

   INPUTS
       iORequest   - pointer to audio request block (struct IOAudio)
               io_Device   - pointer to device node, must be set by (or
                             copied from I/O block set by) open (OpenDevice)
               io_Unit     - bit map of channels to free (ADCMD_FREE) (bits 0
                             thru 3 correspond to channels 0 thru 3)
               ioa_AllocKey- allocation key, used to free channels

   OUTPUTS
       iORequest - pointer to audio request block (struct IOAudio)
               io_Device   - set to -1
               io_Unit     - set to zero
```

```
audio.device/ADCMD_ALLOCATE

   NAME
      ADCMD_ALLOCATE -- allocate a set of audio channels

   FUNCTION
      ADCMD_ALLOCATE is a command that allocates multiple audio channels.
      ADCMD_ALLOCATE takes an array of possible channel combinations
      (ioa_Data) and an allocation precedence (ln_Pri) and tries to allocate
      one of the combinations of channels.

      If the channel combination array is zero length (ioa_Length), the
      allocation succeeds; otherwise, ADCMD_ALLOCATE checks each
      combination, one at a time, in the specified order, to find one
      combination that does not require ADCMD_ALLOCATE to steal allocated
      channels.

      If it must steal allocated channels, it uses the channel combination
      that steals the lowest precedence channels.

      ADCMD_ALLOCATE cannot steal a channel of equal or greater precedence
      than the allocation precedence (ln_Pri).

      If it fails to allocate any channel combination and the no-wait flag
      (ADIOF_NOWAIT) is set ADCMD_ALLOCATE returns a zero in the unit field
      of the I/O request (io_Unit) and an error (IOERR_ALLOCFAILED).  If the
      no-wait flag is clear, it places the I/O request in a list that tries
      to allocate again whenever ADCMD_FREE frees channels or ADCMD_SETPREC
      lowers the channels' precedences.

      If the allocation is successful, ADCMD_ALLOCATE checks if any channels
      are locked (ADCMD_LOCK) and if so, replies (ReplyMsg) the lock I/O
      request with an error (ADIOERR_CHANNELSTOLEN). Then it places the
      allocation I/O request in a list waiting for the locked channels to be
      freed.  When all the allocated channels are un-locked, ADCMD_ALLOCATE:
        . resets (CMD_RESET) the allocated channels,
        . generates a new allocation key (ioa_AllocKey), if it is zero,
        . copies the allocation key into each of the allocated channels
        . copies the allocation precedence into each of the allocated
          channels, and
        . copies the channel bit map into the unit field of the I/O request.

      If channels are allocated with a non-zero allocation key,
      ADCMD_ALLOCATE allocates with that same key; otherwise, it generates a
      new and unique key.

      ADCMD_ALLOCATE is synchronous:
        . if the allocation succeeds and there are no locked channels to be
          stolen, or
        . if the allocation fails and the no-wait flag is set.
       . if the allocation fails and the no-wait flag is set.
      In either case, ADCMD_ALLOCATE only replies (mn_ReplyPort) if the
      quick flag (IOF_QUICK) is clear; otherwise, the allocation is
      asynchronous, so it clears the quick flag and replies the I/O request
      after the allocation is finished.  If channels are stolen, all audio
      device commands return an error (IOERR_NOALLOCATION) when the former
      user tries to use them again.  Do not use ADCMD_ALLOCATE in interrupt
      code.

      If you decide to store directly to the audio hardware registers, you
      must either lock the channels you've allocated, or set the precedence
      to maximum (ADALLOC_MAXPREC) to prevent the channels from being
      stolen.

      Under all circumstances, unless channels are stolen, you must free
```

```
      (ADCMD_FREE) all allocated channels when you are finished using them.

   INPUTS
      ln_Pri       - allocation precedence (-128 thru 127)
      mn_ReplyPort- pointer to message port that receives I/O request after
                     the allocation completes is asynchronous or quick flag
                     (ADIOF_QUICK) is set
      io_Device    - pointer to device node, must be set by (or copied from
                     I/O block set by) OpenDevice function
      io_Command   - command number for ADCMD_ALLOCATE
      io_Flags     - flags, must be cleared if not used:
                     IOF_QUICK    - (CLEAR) reply I/O request
                                    (SET) only reply I/O request only if
                                       asynchronous (see above text)
                     ADIOF_NOWAIT- (CLEAR) if allocation fails, wait till is
                                          succeeds
                                    (SET) if allocation fails, return error
                                          (ADIOERR_ALLOCFAILED)
      ioa_AllocKey- allocation key, zero to generate new key; otherwise,
                     it must be set by (or copied from I/O block set by)
                     OpenDevice function or previous ADCMD_ALLOCATE command
      ioa_Data     - pointer to channel combination options (byte array, bits
                     0 thru 3 correspond to channels 0 thru 3)
      ioa_Length   - length of the channel combination option array
                     (0 thru 16, 0 always succeeds)

   OUTPUTS
      io_Unit      - bit map of successfully allocated channels (bits 0 thru
                     3 correspond to channels 0 thru 3)
      io_Flags     - IOF_QUICK flag cleared if asynchronous (see above text)
      io_Error     - error number:
                     0                   - no error
                     ADIOERR_ALLOCFAILED - allocation failed
      ioa_AllocKey- allocation key, set to a unique number if passed a zero
                     and command succeeds
```

```
audio.device/ADCMD_FINISH                    audio.device/ADCMD_FINISH

   NAME
      ADCMD_FINISH -- abort writes in progress to audio channels

   FUNCTION
      ADCMD_FINISH is a command for multiple audio channels.  For each
      selected channel (io_Unit), if the allocation key (ioa_AllocKey) is
      correct and there is a write (CMD_WRITE)in progress, ADCMD_FINISH
      aborts the current write immediately or at the end of the current
      cycle depending on the sync flag (ADIOF_SYNCCYCLE).  If the allocation
      key is incorrect ADCMD_FINISH returns an error (ADIOERR_NOALLOCATION).
      ADCMD_FINISH is synchronous and only replies (mn_ReplyPort) if the
      quick flag (IOF_QUICK) is clear.  Do not use ADCMD_FINISH in interrupt
      code at interrupt level 5 or higher.

   INPUTS
      mn_ReplyPort- pointer to message port that receives I/O request
                    if the quick flag (IOF_QUICK) is clear
      io_Device   - pointer to device node, must be set by (or copied from
                    I/O block set by) OpenDevice function
      io_Unit     - bit map of channels to finish (bits 0 thru 3 correspond
                    to channels 0 thru 3)
      io_Command  - command number for ADCMD_FINISH
      io_Flags    - flags, must be cleared if not used:
                    IOF_QUICK      - (CLEAR) reply I/O request
                    ADIOF_SYNCCYCLE- (CLEAR) finish immediately
                                     (SET)  finish at the end of current
                                            cycle

      ioa_AllocKey- allocation key, must be set by (or copied from I/O block
                    set by) OpenDevice function or ADCMD_ALLOCATE command

   OUTPUTS
      io_Unit     - bit map of channels successfully finished (bits 0 thru 3
                    correspond to channels 0 thru 3)
      io_Error    - error number:
                    0                    - no error
                    ADIOERR_NOALLOCATION - allocation key (ioa_AllocKey)
                                           does not match key for channel
```

```
audio.device/ADCMD_FREE                      audio.device/ADCMD_FREE

   NAME
      ADCMD_FREE -- free audio channels for allocation

   FUNCTION
      ADCMD_FREE is a command for multiple audio channels.  For each
      selected channel (io_Unit), if the allocation key (ioa_AllocKey) is
      correct, ADCMD_FREE does the following:
        . restores the channel to a known state (CMD_RESET),
        . changes the channels allocation key, and
        . makes the channel available for re-allocation.
        . If the channel is locked (ADCMD_LOCK) ADCMD_FREE unlocks it and
          clears the bit for the channel (io_Unit) in the lock I/O request.
          If the lock I/O request has no channel bits set ADCMD_FREE replies
          the lock I/O request, and
        . checks if there are allocation requests (ADCMD_ALLOCATE) waiting
          for the channel.

      Otherwise, ADCMD_FREE returns an error (ADIOERR_NOALLOCATION).
      ADCMD_FREE is synchronous and only replies (mn_ReplyPort) if the quick
      flag (IOF_QUICK) is clear.  Do not use ADCMD_FREE in interrupt code.

   INPUTS
      mn_ReplyPort- pointer to message port that receives I/O request
                    if the quick flag (IOF_QUICK) is clear
      io_Device   - pointer to device node, must be set by (or copied from
                    I/O block set by) OpenDevice function
      io_Unit     - bit map of channels to free (bits 0 thru 3 correspond to
                    channels 0 thru 3)
      io_Command  - command number for ADCMD_FREE
      io_Flags    - flags, must be cleared if not used:
                    IOF_QUICK - (CLEAR) reply I/O request
      ioa_AllocKey- allocation key, must be set by (or copied from I/O block
                    set by) OpenDevice function or ADCMD_ALLOCATE command

   OUTPUTS
      io_Unit     - bit map of channels successfully freed (bits 0 thru 3
                    correspond to channels 0 thru 3)
      io_Error    - error number:
                    0                    - no error
                    ADIOERR_NOALLOCATION - allocation key (ioa_AllocKey)
                                           does not match key for channel
```

```
audio.device/ADCMD_LOCK                       audio.device/ADCMD_LOCK

   NAME
       ADCMD_LOCK -- prevent audio channels from being stolen

   FUNCTION
       ADCMD_LOCK is a command for multiple audio channels.  For each
       selected channel (io_Unit), if the allocation key (ioa_AllocKey) is
       correct, ADCMD_LOCK locks the channel, preventing subsequent
       allocations (ADCMD_ALLOCATE or OpenDevice) from stealing the channel.
       Otherwise, ADCMD_LOCK returns an error (ADIOERR_NOALLOCATION) and will
       not lock any channels.

       Unlike setting the precedence (ADCMD_SETPREC, ADCMD_ALLOCATE or
       OpenDevice) to maximum (ADALLOC_MAXPREC) which would cause all
       subsequent allocations to fail, ADCMD_LOCK causes all higher
       precedence allocations, even no-wait (ADIOF_NOWAIT) allocations, to
       wait until the channels are un-locked.

       Locked channels can only be unlocked by freeing them (ADCMD_FREE),
       which clears the channel select bits (io_Unit).  ADCMD_LOCK does not
       reply the I/O request (mn_ReplyPort) until all the channels it locks
       are freed, unless a higher precedence allocation attempts to steal one
       the locked channels. If a steal occurs, ADCMD_LOCK replies and returns
       an error (ADIOERR_CHANNELSTOLEN).  If the lock is replied
       (mn_ReplyPort) with this error, the channels should be freed as soon
       as possible.  To avoid a possible deadlock, never make the freeing of
       stolen channels dependent on another allocations completion.

       ADCMD_LOCK is only asynchronous if the allocation key is correct, in
       which case it clears the quick flag (IOF_QUICK); otherwise, it is
       synchronous and only replies if the quick flag (IOF_QUICK) is clear.
       Do not use ADCMD_LOCK in interrupt code.

   INPUTS
       mn_ReplyPort- pointer to message port that receives I/O request
                     if the quick flag (IOF_QUICK) is clear
       io_Device   - pointer to device node, must be set by (or copied from
                     I/O block set by) OpenDevice function
       io_Unit     - bit map of channels to lock (bits 0 thru 3 correspond to
                     channels 0 thru 3)
       io_Command  - command number for ADCMD_LOCK
       io_Flags    - flags, must be cleared
       ioa_AllocKey- allocation key, must be set by (or copied from I/O block
                     set by) OpenDevice function or ADCMD_ALLOCATE command

   OUTPUTS
       io_Unit     - bit map of successfully locked channels (bits 0 thru 3
                     correspond to channels 0 thru 3) not freed (ADCMD_FREE)
       io_Flags    - IOF_QUICK flag cleared if the allocation key is correct
                     (no ADIOERR_NOALLOCATION error)
       io_Error    - error number:
                     0                    - no error
                     ADIOERR_NOALLOCATION - allocation key (ioa_AllocKey)
                                            does not match key for channel
                     ADIOERR_CHANNELSTOLEN- allocation attempting to steal
                                            locked channel
```

```
audio.device/ADCMD_PERVOL                     audio.device/ADCMD_PERVOL

   NAME
       ADCMD_PERVOL -- change the period and volume for writes in progress to
                       audio channels

   FUNCTION
       ADCMD_PERVOL is a command for multiple audio channels.  For each
       selected channel (io_Unit), if the allocation key (ioa_AllocKey) is
       correct and there is a write (CMD_WRITE) in progress, ADCMD_PERVOL
       loads a new volume and period immediately or at the end of the current
       cycle depending on the sync flag (ADIOF_SYNCCYCLE).  If the allocation
       key in incorrect, ADCMD_PERVOL returns an error
       (ADIOERR_NOALLOCATION).  ADCMD_PERVOL is synchronous and only replies
       (mn_ReplyPort) if the quick flag (IOF_QUICK) is clear.  Do not use
       ADCMD_PERVOL in interrupt code at interrupt level 5 or higher.

   INPUTS
       mn_ReplyPort- pointer to message port that receives I/O request
                     if the quick flag (IOF_QUICK) is clear
       io_Device   - pointer to device node, must be set by (or copied from
                     I/O block set by) OpenDevice function
       io_Unit     - bit map of channels to load period and volume (bits 0
                     thru 3 correspond to channels 0 thru 3)
       io_Command  - command number for ADCMD_PERVOL
       io_Flags    - flags, must be cleared if not used:
                     IOF_QUICK       - (CLEAR) reply I/O request
                     ADIOF_SYNCCYCLE- (CLEAR) load period and volume
                                              immediately
                                      (SET) load period and volume at the end
                                              of the current cycle
       ioa_AllocKey- allocation key, must be set by (or copied from I/O block
                     set by) OpenDevice function or ADCMD_ALLOCATE command
       ioa_Period  - new sample period in 279.365 ns increments (124 thru
                     65536, anti-aliasing filter works below 300 to 500
                     depending on waveform)
       ioa_Volume  - new volume (0 thru 64, linear)

   OUTPUTS
       io_Unit     - bit map of channels that successfully loaded period and
                     volume (bits 0 thru 3 correspond to channels 0 thru 3)
       io_Error    - error number:
                     0                    - no error
                     ADIOERR_NOALLOCATION - allocation key (ioa_AllocKey)
                                            does not match key for channel
```

```
audio.device/ADCMD_SETPREC                audio.device/ADCMD_SETPREC

   NAME
       ADCMD_SETPREC -- set the allocation precedence for audio channels

   FUNCTION
       ADCMD_SETPREC is a command for multiple audio channels.  For each
       selected channel (io_Unit), if the allocation key (ioa_AllocKey) is
       correct, ADCMD_SETPREC sets the allocation precedence to a new value
       (ln_Pri) and checks if there are allocation requests (ADCMD_ALLOCATE)
       waiting for the channel which now have higher precedence; otherwise,
       ADCMD_SETPREC returns an error (ADIOERR_NOALLOCATION).  ADCMD_SETPREC
       is synchronous and only replies (mn_ReplyPort) if the quick flag
       (IOF_QUICK) is clear.  Do not use ADCMD_SETPREC in interrupt code.

   INPUTS
       ln_Pri       - new allocation precedence (-128 thru 127)
       mn_ReplyPort- pointer to message port that receives I/O request
                      if the quick flag (IOF_QUICK) is clear
       io_Device    - pointer to device node, must be set by (or copied from
                      I/O block set by) OpenDevice function
       io_Unit      - bit map of channels to set precedence (bits 0 thru 3
                      correspond to channels 0 thru 3)
       io_Command   - command number for ADCMD_SETPREC
       io_Flags     - flags, must be cleared if not used:
                      IOF_QUICK - (CLEAR) reply I/O request
       ioa_AllocKey- allocation key, must be set by (or copied from I/O block
                      set by) OpenDevice function or ADCMD_ALLOCATE command

   OUTPUTS
       io_Unit      - bit map of channels that successfully set precedence
                      (bits 0 thru 3 correspond to channels 0 thru 3)
       io_Error     - error number:
                      0                    - no error
                      ADIOERR_NOALLOCATION - allocation key (ioa_AllocKey)
                                             does not match key for channel
```

```
audio.device/ADCMD_WAITCYCLE

   NAME
       ADCMD_WAITCYCLE -- wait for an audio channel to complete the current
                          cycle of a write

   FUNCTION
       ADCMD_WAITCYCLE is a command for a single audio channel (io_Unit).
       If the allocation key (ioa_AllocKey) is correct and there is a write
       (CMD_WRITE) in progress on selected channel, ADCMD_WAITCYCLE does not
       reply (mn_ReplyPort) until the end of the current cycle.  If there is
       no write is progress, ADCMD_WAITCYCLE replies immediately.  If the
       allocation key is incorrect, ADCMD_WAITCYCLE returns an error
       (ADIOERR_NOALLOCATION).  ADCMD_WAITCYCLE returns an error
       (IOERR_ABORTED) if it is canceled (AbortIO) or the channel is stolen
       (ADCMD_ALLOCATE).  ADCMD_WAITCYCLE is only asynchronous if it is
       waiting for a cycle to complete, in which case it clears the quick
       flag (IOF_QUICK); otherwise, it is synchronous and only replies if the
       quick flag (IOF_QUICK) is clear.  Do not use ADCMD_WAITCYCLE in
       interrupt code at interrupt level 5 or higher.

   INPUTS
       mn_ReplyPort- pointer to message port that receives I/O request, if
                      the quick flag (IOF_QUICK) is clear, or if a write is in
                      progress on the selected channel and a cycle has
                      completed
       io_Device    - pointer to device node, must be set by (or copied from
                      I/O block set by) OpenDevice function
       io_Unit      - bit map of channel to wait for cycle (bits 0 thru 3
                      correspond to channels 0 thru 3), if more then one bit
                      is set lowest bit number channel is used
       io_Command   - command number for CMD_WAITCYCLE
       io_Flags     - flags, must be cleared if not used:
                      IOF_QUICK - (CLEAR) reply I/O request
                                  (SET) only reply I/O request if a write is
                                        in progress on the selected channel
                                        and a cycle has completed
       ioa_AllocKey- allocation key, must be set by (or copied from I/O block
                      set by) OpenDevice function or ADCMD_ALLOCATE command

   OUTPUTS
       io_Unit      - bit map of channel that successfully waited for cycle
                      (bits 0 thru 3 correspond to channels 0 thru 3)
       io_Flags     - IOF_QUICK flag cleared if a write is in progress on the
                      selected channel
       io_Error     - error number:
                      0                    - no error
                      IOERR_ABORTED        - canceled (AbortIO) or channel
                                             stolen
                      ADIOERR_NOALLOCATION - allocation key (ioa_AllocKey)
                                             does not match key for channel
```

audio.device Page 13

```
audio.device/CMD_CLEAR                    audio.device/CMD_CLEAR

   NAME
       CMD_CLEAR -- throw away internal caches

   FUNCTION
       CMD_CLEAR is a standard command for multiple audio channels.  For each
       selected channel (io_Unit), if the allocation key (ioa_AllocKey) is
       correct, CMD_CLEAR does nothing; otherwise, CMD_CLEAR returns an error
       (ADIOERR_NOALLOCATION).  CMD_CLEAR is synchronous and only replies
       (mn_ReplyPort) if the quick flag (IOF_QUICK) is clear.

   INPUTS
       mn_ReplyPort- pointer to message port that receives I/O request after
                     if the quick flag (IOF_QUICK) is clear
       io_Device   - pointer to device node, must be set by (or copied from
                     I/O block set by) OpenDevice function
       io_Unit     - bit map of channels to clear (bits 0 thru 3 correspond
                     to channels 0 thru 3)
       io_Command  - command number for CMD_CLEAR
       io_Flags    - flags, must be cleared if not used:
                     IOF_QUICK - (CLEAR) reply I/O request
       ioa_AllocKey- allocation key, must be set by (or copied from I/O block
                     set by) OpenDevice function or ADCMD_ALLOCATE command

   OUTPUTS
       io_Unit     - bit map of channels successfully cleared (bits 0 thru 3
                     correspond to channels 0 thru 3)
       io_Error    - error number:
                     0                    - no error
                     ADIOERR_NOALLOCATION - allocation key (ioa_AllocKey)
                                            does not match key for channel
```

audio.device Page 14

```
audio.device/CMD_FLUSH                    audio.device/CMD_FLUSH

   NAME
       CMD_FLUSH -- cancel all pending I/O

   FUNCTION
       CMD_FLUSH is a standard command for multiple audio channels.  For each
       selected channel (io_Unit), if the allocation key (ioa_AllocKey) is
       correct, CMD_FLUSH aborts all writes (CMD_WRITE) in progress or queued
       and any I/O requests waiting to synchronize with the end of the cycle
       (ADCMD_WAITCYCLE); otherwise, CMD_FLUSH returns an error
       (ADIOERR_NOALLOCATION).  CMD_FLUSH is synchronous and only replies
       (mn_ReplyPort) if the quick flag (IOF_QUICK) is clear.  Do not use
       CMD_FLUSH in interrupt code at interrupt level 5 or higher.

   INPUTS
       mn_ReplyPort- pointer to message port that receives I/O request
                     if the quick flag (IOF_QUICK) is clear
       io_Device   - pointer to device node, must be set by (or copied from
                     I/O block set by) OpenDevice function
       io_Unit     - bit map of channels to flush (bits 0 thru 3 correspond
                     to channels 0 thru 3)
       io_Command  - command number for CMD_FLUSH
       io_Flags    - flags, must be cleared if not used:
                     IOF_QUICK - (CLEAR) reply I/O request
       ioa_AllocKey- allocation key, must be set by (or copied from I/O block
                     set by) OpenDevice function or ADCMD_ALLOCATE command

   OUTPUTS
       io_Unit     - bit map of channels successfully flushed (bits 0 thru 3
                     correspond to channels 0 thru 3)
       io_Error    - error number:
                     0                    - no error
                     ADIOERR_NOALLOCATION - allocation key (ioa_AllocKey)
                                            does not match key for channel
```

```
audio.device/CMD_READ                              audio.device/CMD_READ

   NAME
       CMD_READ -- normal I/O entry point

   FUNCTION
       CMD_READ is a standard command for a single audio channel (io_Unit).
       If the allocation key (ioa_AllocKey) is correct, CMD_READ returns a
       pointer (io_Data) to the I/O block currently writing (CMD_WRITE) on
       the selected channel; otherwise, CMD_READ returns an error
       (ADIOERR_NOALLOCATION).  If there is no write in progress, CMD_READ
       returns zero.  CMD_READ is synchronous and only replies (mn_ReplyPort)
       if the quick bit (IOF_QUICK) is clear.

   INPUTS
       mn_ReplyPort- pointer to message port that receives I/O request after
                     if the quick flag (IOF_QUICK) is clear
       io_Device   - pointer to device node, must be set by (or copied from
                     I/O block set by) OpenDevice function
       io_Unit     - bit map of channel to read (bit 0 thru 3 corresponds to
                     channel 0 thru 3), if more then one bit is set lowest
                     bit number channel read
       io_Command  - command number for CMD_READ
       io_Flags    - flags, must be cleared if not used:
                     IOF_QUICK - (CLEAR) reply I/O request
       ioa_AllocKey- allocation key, must be set by (or copied from I/O block
                     set by) OpenDevice function or ADCMD_ALLOCATE command

   OUTPUTS
       io_Unit     - bit map of channel successfully read (bit 0 thru 3
                     corresponds to channel 0 thru 3)
       io_Error    - error number:
                     0                    - no error
                     ADIOERR_NOALLOCATION - allocation key (ioa_AllocKey)
                                            does not match key for channel
       ioa_Data    - pointer to I/O block for current write, zero if none is
                     progress
```

```
audio.device/CMD_RESET                              audio.device/CMD_RESET

   NAME
       CMD_RESET -- restore device to a known state

   FUNCTION
       CMD_RESET is a standard command for multiple audio channels.  For each
       selected channel (io_Unit), if the allocation key (ioa_AllocKey) is
       correct, CMD_RESET:
         . clears the hardware audio registers and attach bits,
         . sets the audio interrupt vector,
         . cancels all pending I/O (CMD_FLUSH), and
         . un-stops the channel if it is stopped (CMD_STOP),

       Otherwise, CMD_RESET returns an error (ADIOERR_NOALLOCATION).
       CMD_RESET is synchronous and only replies (mn_ReplyPort) if the quick
       flag (IOF_QUICK) is clear.  Do not use CMD_RESET in interrupt code at
       interrupt level 5 or higher.

   INPUTS
       mn_ReplyPort- pointer to message port that receives I/O request
                     if the quick flag (IOF_QUICK) is clear
       io_Device   - pointer to device node, must be set by (or copied from
                     I/O block set by) OpenDevice function
       io_Unit     - bit map of channels to reset (bits 0 thru 3 correspond
                     to channels 0 thru 3)
       io_Command  - command number for CMD_RESET
       io_Flags    - flags, must be cleared if not used:
                     IOF_QUICK - (CLEAR) reply I/O request
       ioa_AllocKey- allocation key, must be set by (or copied from I/O block
                     set by) OpenDevice function or ADCMD_ALLOCATE command

   OUTPUTS
       io_Unit     - bit map of channels to successfully reset (bits 0 thru 3
                     correspond to channels 0 thru 3)
       io_Error    - error number:
                     0                    - no error
                     ADIOERR_NOALLOCATION - allocation key (ioa_AllocKey)
                                            does not match key for channel
```

```
audio.device/CMD_START                            audio.device/CMD_START

   NAME
       CMD_START -- start device processing (like ^Q)

   FUNCTION
       CMD_START is a standard command for multiple audio channels.  For each
       selected channel (io_Unit), if the allocation key (ioa_AllocKey) is
       correct and the channel was previously stopped (CMD_STOP), CMP_START
       immediately starts all writes (CMD_WRITE) to the channel.  If the
       allocation key is incorrect, CMD_START returns an error
       (ADIOERR_NOALLOCATION).  CMD_START starts multiple channels
       simultaneously to minimize distortion if the channels are playing the
       same waveform and their outputs are mixed. CMD_START is synchronous and

       only replies (mn_ReplyPort) if the quick flag (IOF_QUICK) is clear.  Do

       not use CMD_START in interrupt code at interrupt level 5 or higher.

   INPUTS
       mn_ReplyPort- pointer to message port that receives I/O request after
                     if the quick flag (IOF_QUICK) is clear
       io_Device   - pointer to device node, must be set by (or copied from
                     I/O block set by) OpenDevice function
       io_Unit     - bit map of channels to start (bits 0 thru 3 correspond
                     to channels 0 thru 3)
       io_Command  - command number for CMD_START
       io_Flags    - flags, must be cleared if not used:
                     IOF_QUICK - (CLEAR) reply I/O request
       ioa_AllocKey- allocation key, must be set by (or copied from I/O block
                     set by) OpenDevice function or ADCMD_ALLOCATE command

   OUTPUTS
       io_Unit     - bit map of channels successfully started (bits 0 thru 3
                     correspond to channels 0 thru 3)
       io_Error    - error number:
                     0                    - no error
                     ADIOERR_NOALLOCATION - allocation key (ioa_AllocKey)
                                            does not match key for channel
```

```
audio.device/CMD_STOP                             audio.device/CMD_STOP

   NAME
       CMD_STOP -- stop device processing (like ^S)

   FUNCTION
       CMD_STOP is a standard command for multiple audio channels.  For each
       selected channel (io_Unit), if the allocation key (ioa_AllocKey) is
       correct, CMD_STOP immediately stops any writes (CMD_WRITE) in
       progress; otherwise, CMD_STOP returns an error (ADIOERR_NOALLOCATION).
       CMD_WRITE queues up writes to a stopped channel until CMD_START starts
       the channel or CMD_RESET resets the channel. CMD_STOP is synchronous
       and only replies (mn_ReplyPort) if the quick flag (IOF_QUICK) is
       clear.  Do not use CMD_STOP in interrupt code at interrupt level 5 or
       higher.

   INPUTS
       mn_ReplyPort- pointer to message port that receives I/O request after
                     if the quick flag (IOF_QUICK) is clear
       io_Device   - pointer to device node, must be set by (or copied from
                     I/O block set by) OpenDevice function
       io_Unit     - bit map of channels to stop (bits 0 thru 3 correspond to
                     channels 0 thru 3)
       io_Command  - command number for CMD_STOP
       io_Flags    - flags, must be cleared if not used:
                     IOF_QUICK - (CLEAR) reply I/O request
       ioa_AllocKey- allocation key, must be set by (or copied from I/O block
                     set by) OpenDevice function or ADCMD_ALLOCATE command

   OUTPUTS
       io_Unit     - bit map of channels successfully stopped (bits 0 thru 3
                     correspond to channels 0 thru 3)
       io_Error    - error number:
                     0                    - no error
                     ADIOERR_NOALLOCATION - allocation key (ioa_AllocKey)
                                            does not match key for channel
```

```
audio.device/CMD_UPDATE                        audio.device/CMD_UPDATE

   NAME
       CMD_UPDATE -- force dirty buffers out

   FUNCTION
       CMD_UPDATE is a standard command for multiple audio channels.  For
       each selected channel (io_Unit), if the allocation key (ioa_AllocKey)
       is correct, CMD_UPDATE does nothing; otherwise, CMD_UPDATE returns an
       error (ADIOERR_NOALLOCATION).  CMD_UPDATE is synchronous and only
       replies (mn_ReplyPort) if the quick flag (IOF_QUICK) is clear.

   INPUTS
       mn_ReplyPort- pointer to message port that receives I/O request after
                     if the quick flag (IOF_QUICK) is clear
       io_Device   - pointer to device node, must be set by (or copied from
                     I/O block set by) OpenDevice function
       io_Unit     - bit map of channels to update (bits 0 thru 3 correspond
                     to channels 0 thru 3)
       io_Command  - command number for CMD_UPDATE
       io_Flags    - flags, must be cleared if not used:
                     IOF_QUICK - (CLEAR) reply I/O request
       ioa_AllocKey- allocation key, must be set by (or copied from I/O block
                     set by) OpenDevice function or ADCMD_ALLOCATE command

   OUTPUTS
       io_Unit     - bit map of channels successfully updated (bits 0 thru 3
                     correspond to channels 0 thru 3)
       io_Error    - error number:
                     0                    - no error
                     ADIOERR_NOALLOCATION - allocation key (ioa_AllocKey)
                                            does not match key for channel
```

```
audio.device/CMD_WRITE                         audio.device/CMD_WRITE

   NAME
       CMD_WRITE -- normal I/O entry point

   FUNCTION
       CMD_WRITE is a standard command for a single audio channel (io_Unit).
       If the allocation key (ioa_AllocKey) is correct, CMD_WRITE plays a
       sound using the selected channel; otherwise, it returns an error
       (ADIOERR_NOALLOCATION).  CMD_WRITE queues up requests if there is
       another write in progress or if the channel is stopped (CMD_STOP).
       When the write actually starts; if the ADIOF_PERVOL flag is set,
       CMD_WRITE loads volume (ioa_Volume) and period (ioa_Period), and if
       the ADIOF_WRITEMESSAGE flag is set, CMD_WRITE replies the write
       message (ioa_WriteMsg).  CMD_WRITE returns an error (IOERR_ABORTED) if
       it is canceled (AbortIO) or the channel is stolen (ADCMD_ALLOCATE).
       CMD_WRITE is only asynchronous if there is no error, in which case it
       clears the quick flag (IOF_QUICK) and replies the I/O request
       (mn_ReplyPort) after it finishes writting; otherwise, it is synchronous

       and only replies if the quick flag (IOF_QUICK) is clear.  Do not use
       CMD_WRITE in interrupt code at interrupt level 5 or higher.

   INPUTS
       mn_ReplyPort- pointer to message port that receives I/O request after
                     the write completes
       io_Device   - pointer to device node, must be set by (or copied from
                     I/O block set by) OpenDevice function
       io_Unit     - bit map of channel to write (bit 0 thru 3 corresponds to
                     channel 0 thru 3), if more then one bit is set lowest
                     bit number channel is written
       io_Command  - command number for CMD_WRITE
       io_Flags    - flags, must be cleared if not used:
                     ADIOF_PERVOL       - (SET) load volume and period
                     ADIOF_WRITEMESSAGE - (SET) reply message at write start
       ioa_AllocKey- allocation key, must be set by (or copied from I/O block
                     set by) OpenDevice function or ADCMD_ALLOCATE command
       ioa_Data    - pointer to waveform array (signed bytes (-128 thru 127)
                     in custom chip addressable ram and word aligned)
       ioa_Length  - length of the wave array in bytes (2 thru 131072, must
                     be even number)
       ioa_Period  - sample period in 279.365 ns increments (124 thru 65536,
                     anti-aliasing filter works below 300 to 500 depending on
                     waveform), if enabled by ADIOF_PERVOL
       ioa_Volume  - volume (0 thru 64, linear), if enabled by ADIOF_PERVOL
       ioa_Cycles  - number of times to repeat array (0 thru 65535, 0 for
                     infinite)
       ioa_WriteMsg- message replied at start of write,  if enabled by
                     ADIOF_WRITEMESSAGE

   OUTPUTS
       io_Unit     - bit map of channel successfully written (bit 0 thru 3
                     corresponds to channel 0 thru 3)
       io_Flags    - IOF_QUICK flag cleared if there is no error
       io_Error    - error number:
                     0                    - no error
                     IOERR_ABORTED        - canceled (AbortIO) or channel
                                            stolen
                     ADIOERR_NOALLOCATION - allocation key (ioa_AllocKey)
                                            does not match key for channel

   BUGS
       If CMD_WRITE starts the write immediately after stopping a previous
       write, you must set the ADIOF_PERVOL flag or else the new data pointer
       (ioa_Data) and length (ioa_Length) may not be loaded.
```

```
audio.device/Expunge                                   audio.device/Expunge

   NAME
       EXPUNGE - indicate a desire to remove the Audio device

   FUNCTION
       The Expunge routine is called when a user issues a RemDevice call.  By
       the time it is called, the device has already been removed from the
       device list, so no new opens will succeed.  The existence of any other
       users of the device, as determined by the device open count being
       non-zero, will cause the Expunge to be deferred.  When the device is
       not in use, or no longer in use, the Expunge is actually performed.
```

```
audio.device/OpenDevice                               audio.device/OpenDevice

   NAME
       OpenDevice - open the audio device

   SYNOPSIS
       error = OpenDevice("audio.device", unitNumber, iORequest, flags);

   FUNCTION
       The OpenDevice routine grants access to the audio device.  It takes an
       I/O audio request block (iORequest) and if it can successfully open
       the audio device, it loads the device pointer (io_Device) and the
       allocation key (ioa_AllocKey); otherwise, it returns an error
       (IOERR_OPENFAIL).  OpenDevice increments the open count keeping the
       device from being expunged (Expunge).  If the length (ioa_Length) is
       non-zero, OpenDevice tries to allocate (ADCMD_ALLOCATE) audio channels
       from a array of channel combination options (ioa_Data). If the
       allocation succeeds, the allocated channel combination is loaded into
       the unit field (ioa_Unit); otherwise, OpenDevice returns an error
       (ADIOERR_ALLOCFAILED).  OpenDevice does not wait for allocation to
       succeed and closes (CloseDevice) the audio device if it fails.  To
       allocate channels, OpenDevice also requires a properly initialized
       reply port (mn_ReplyPort) with an allocated signal bit.

   INPUTS
       unitNumber- not used
       iORequest - pointer to audio request block (struct IOAudio)
               ln_Pri      - allocation precedence (-128 thru 127), only
                             necessary for allocation (non-zero length)
               mn_ReplyPort- pointer to message port for allocation, only
                             necessary for allocation (non-zero length)
               ioa_AllocKey- allocation key; zero to generate new key.
                             Otherwise, it must be set by (or copied from I/O
                             block that is set by) previous OpenDevice
                             function or ADCMD_ALLOCATE command (non-zero
                             length)
               ioa_Data    - pointer to channel combination options (byte
                             array, bits 0 thru 3 correspond to channels 0
                             thru 3), only necessary for allocation (non-zero
                             length)
               ioa_Length  - length of the channel combination option array
                             (0 thru 16), zero for no allocation
       flags     - not used

   OUTPUTS
       iORequest - pointer to audio request block (struct IOAudio)
               io_Device   - pointer to device node if OpenDevice succeeds,
                             otherwise -1
               io_Unit     - bit map of successfully allocated channels (bits
                             0 thru 3 correspond to channels 0 thru 3)
               io_Error    - error number:
                             0                   - no error
                             IOERR_OPENFAIL      - open failed
                             ADIOERR_ALLOCFAILED - allocation failed, no open
               ioa_AllocKey- allocation key, set to a unique number if passed
                             a zero and OpenDevice succeeds
       error     - copy of io_Error
```

clipboard.device Page 1

TABLE OF CONTENTS

clipboard.device/CBD_CHANGEHOOK
clipboard.device/CBD_CURRENTREADID
clipboard.device/CBD_CURRENTWRITEID
clipboard.device/CBD_POST
clipboard.device/CMD_READ
clipboard.device/CMD_UPDATE
clipboard.device/CMD_WRITE

clipboard.device Page 2

```
clipboard.device/CBD_CHANGEHOOK               clipboard.device/CBD_CHANGEHOOK

   NAME
        CBD_CHANGEHOOK -- Add or remove a clip change hook

   FUNCTION
        CBD_CHANGEHOOK allows specification of a hook to be called
        when the data on the clipboard has changed.

   IO REQUEST
        io_Message      mn_ReplyPort set up
        io_Device       preset by OpenDevice
        io_Unit         preset by OpenDevice
        io_Command      CBD_CHANGEHOOK
        io_Length -     0 to remove, 1 to install this hook
        io_Data -       struct Hook *, the clip change hook

   HOOK ENVIRONMENT
       hook message - a ClipHookMsg, as defined in devices/clipboard.h
           chm_Type - 0, indicating that the message has the
               following fields:
           chm_ClipID - the clip ID of the clip triggering the change
       hook object - io_Unit
```

clipboard.device Page 3

```
clipboard.device/CBD_CURRENTREADID          clipboard.device/CBD_CURRENTREADID

   NAME
       CBD_CURRENTREADID - Determine the current read identifier.

   FUNCTION
       CBD_CURRENTREADID fills the io_ClipID with a clip identifier that
       can be compared with that of a post command: if greater than
       the post identifier then the post data held privately by an
       application is not valid for its own pasting.

   IO REQUEST
       io_Message       mn_ReplyPort set up
       io_Device        preset by OpenDevice
       io_Unit          preset by OpenDevice
       io_Command       CBD_CURRENTREADID

   RESULTS
       io_ClipID        the ClipID of the current write is set
```

clipboard.device Page 4

```
clipboard.device/CBD_CURRENTWRITEID        clipboard.device/CBD_CURRENTWRITEID

   NAME
       CBD_CURRENTWRITEID -- Determine the current write identifier.

   FUNCTION
       CBD_CURRENTWRITEID fills the io_ClipID with a clip identifier that
       can be compared with that of a post command: if greater than
       the post identifier then the post is obsolete and need never
       be satisfied.

   IO REQUEST
       io_Message       mn_ReplyPort set up
       io_Device        preset by OpenDevice
       io_Unit          preset by OpenDevice
       io_Command       CBD_CURRENTWRITEID

   RESULTS
       io_ClipID        the ClipID of the current write is set
```

```
clipboard.device/CBD_POST                              clipboard.device/CBD_POST

   NAME
        CBD_POST -- Post availability of a clip to the clipboard.

   FUNCTION
        Indicate to the clipboard device that data is available for
        use by accessors of the clipboard.  This is intended to be
        used when a cut is large, in a private data format, and/or
        changing frequently, and it thus makes sense to avoid
        converting it to an IFF form and writing it to the clipboard
        unless another application wants it.  The post provides a
        message port to which the clipboard device will send a satisfy
        message if the data is required.

        If the satisfy message is received, the write associated with
        the post must be performed.  The act of writing the clip
        indicates that the message has been received: it may then be
        re-used by the clipboard device, and so must actually be
        removed from the satisfy message port so that the port is not
        corrupted.

        If the application wishes to determine if a post it has
        performed is still the current clip, it should check the
        post's io_ClipID with that returned by the CBD_CURRENTREADID
        command.  If the current read io_ClipID is greater, the clip
        is not still current.

        If an application has a pending post and wishes to determine
        if it should satisfy it (e.g. before it exits), it should
        check the post's io_ClipID with that returned by the
        CBD_CURRENTWRITEID command.  If the current write io_ClipID is
        greater, there is no need to satisfy the post.

   IO REQUEST
        io_Message      mn_ReplyPort set up
        io_Device       preset by OpenDevice
        io_Unit         preset by OpenDevice
        io_Command      CBD_POST
        io_Data         pointer to satisfy message port
        io_ClipID       zero

   RESULTS
        io_Error        non-zero if an error occurred
        io_ClipID       the clip ID assigned to this post, to be used
                        in the write command if this is satisfied
```

```
clipboard.device/CMD_READ                              clipboard.device/CMD_READ

   NAME
        CMD_READ -- Read from a clip on the clipboard.

   FUNCTION
        The read function serves two purposes.

        When io_Offset is within the clip, this acts as a normal read
        request, and io_Data is filled with data from the clipboard.
        The first read request should have a zero io_ClipID, which
        will be filled with the ID assigned for this read.  Normal
        sequential access from the beginning of the clip is achieved
        by setting io_Offset to zero for the first read, then leaving
        it untouched for subsequent reads.  If io_Data is null, then
        io_Offset is incremented by io_Actual as if io_Length bytes
        had been read: this is useful to skip to the end of file
        by using a huge io_Length.

        When io_Offset is beyond the end of the clip, this acts as a
        signal to the clipboard device that the application is
        through reading this clip.  Realize that while an application
        is in the middle of reading a clip, any attempts to write new
        data to the clipboard are held off.  This read past the end
        of file indicates that those operations may now be initiated.

   IO REQUEST
        io_Message      mn_ReplyPort set up
        io_Device       preset by OpenDevice
        io_Unit         preset by OpenDevice
        io_Command      CMD_READ
        io_Length       number of bytes to put in data buffer
        io_Data         pointer to buffer of data to fill, or null to
                        skip over data
        io_Offset       byte offset of data to read
        io_ClipID       zero if this is the initial read

   RESULTS
        io_Error        non-zero if an error occurred
        io_Actual       filled with the actual number of bytes read
        io_Data         (the buffer now has io_Actual bytes of data)
        io_Offset       updated to next read position, which is
                        beyond EOF if io_Actual != io_Length
        io_ClipID       the clip ID assigned to this read: do not
                        alter for subsequent reads
```

clipboard.device Page 7

```
clipboard.device/CMD_UPDATE                         clipboard.device/CMD_UPDATE

   NAME
        CMD_UPDATE -- Terminate the writing of a clip to the clipboard.

   FUNCTION
        Indicate to the clipboard that the previous write commands are
        complete and can be used for any pending pastes (reads).  This
        command cannot be issued while any of the write commands are
        pending.

   IO REQUEST
        io_Message      mn_ReplyPort set up
        io_Device       preset by OpenDevice
        io_Unit         preset by OpenDevice
        io_Command      CMD_UPDATE
        io_ClipID       the ClipID of the write

   RESULTS
        io_Error        non-zero if an error occurred
```

clipboard.device Page 8

```
clipboard.device/CMD_WRITE                              clipboard.device/CMD_WRITE

   NAME
        CMD_WRITE -- Write to a clip on the clipboard.

   FUNCTION
        This command writes data to the clipboard.  This data can be
        provided sequentially by clearing io_Offset for the initial
        write, and using the incremented value unaltered for
        subsequent writes.  If io_Offset is ever beyond the current
        clip size, the clip is padded with zeros.

        If this write is in response to a SatisfyMsg for a pending
        post, then the io_ClipID returned by the CBD_POST command must
        be used.  Otherwise, a new ID is obtained by clearing the
        io_ClipID for the first write.  Subsequent writes must not
        alter the io_ClipID.

   IO REQUEST
        io_Message      mn_ReplyPort set up
        io_Device       preset by OpenDevice
        io_Unit         preset by OpenDevice
        io_Command      CMD_WRITE
        io_Length       number of bytes from io_Data to write
        io_Data         pointer to block of data to write
        io_Offset       usually zero if this is the initial write
        io_ClipID       zero if this is the initial write, ClipID of
                        the Post if this is to satisfy a post

   RESULTS
        io_Error        non-zero if an error occurred
        io_Actual       filled with the actual number of bytes written
        io_Offset       updated to next write position
        io_ClipID       the clip ID assigned to this write: do not
                        alter for subsequent writes
```

TABLE OF CONTENTS

console.device/CD_ASKDEFAULTKEYMAP
console.device/CD_ASKKEYMAP
console.device/CD_SETDEFAULTKEYMAP
console.device/CD_SETKEYMAP
console.device/CDInputHandler
console.device/CMD_CLEAR
console.device/CMD_READ
console.device/CMD_WRITE
console.device/OpenDevice
console.device/RawKeyConvert

```
console.device/CD_ASKDEFAULTKEYMAP          console.device/CD_ASKDEFAULTKEYMAP

   NAME
       CD_ASKDEFAULTKEYMAP -- get the current default keymap

   FUNCTION
       Fill the io_Data buffer with the current console device
       default keymap, which is used to initialize console unit
       keymaps when opened, and by RawKeyConvert with a null
       keyMap parameter.

  IO REQUEST
       io_Message      mn_ReplyPort set if quick I/O is not possible
       io_Device       preset by the call to OpenDevice
       io_Unit         preset by the call to OpenDevice
       io_Command      CD_ASKDEFAULTKEYMAP
       io_Flags        IOF_QUICK if quick I/O possible, else zero
       io_Length       sizeof(*keyMap)
       io_Data         struct KeyMap *keyMap
                       pointer to a structure that describes
                       the raw keycode to byte stream conversion.

   RESULTS
       This function sets the io_Error field in the IOStdReq, and fills
       the structure pointed to by io_Data with the current device
       default key map.

   BUGS

   SEE ALSO
       exec/io.h, devices/keymap.h, devices/console.h
```

console.device

Page 3

```
console.device/CD_ASKKEYMAP                          console.device/CD_ASKKEYMAP

   NAME
       CD_ASKKEYMAP -- Get the current key map structure for this console.

    FUNCTION
       Fill the io_Data buffer with the current KeyMap structure in
       use by this console unit.

    IO REQUEST INPUT
       io_Message      mn_ReplyPort set if quick I/O is not possible
       io_Device       preset by the call to OpenDevice
       io_Unit         preset by the call to OpenDevice
       io_Command      CD_ASKKEYMAP
       io_Flags        IOF_QUICK if quick I/O possible, else zero
       io_Length       sizeof(*keyMap)
       io_Data         struct KeyMap *keyMap
                       pointer to a structure that describes
                       the raw keycode to byte stream conversion.

    IO REQUEST RESULT
       This function sets the io_Error field in the IOStdReq, and fills
       the structure the structure pointed to by io_Data with the current
        key map.

    SEE ALSO
       exec/io.h, devices/keymap.h, devices/console.h
```

console.device

Page 4

```
console.device/CD_SETDEFAULTKEYMAP            console.device/CD_SETDEFAULTKEYMAP

   NAME
       CD_SETDEFAULTKEYMAP -- set the current default keymap

   FUNCTION
       This console command copies the keyMap structure pointed to
       by io_Data to the console device default keymap, which is used
       to initialize console units when opened, and by RawKeyConvert
       with a null keyMap parameter.

   IO REQUEST
       io_Message      mn_ReplyPort set if quick I/O is not possible
       io_Device       preset by the call tc OpenDevice
       io_Unit         preset by the call to OpenDevice
       io_Command      CD_SETDEFAULTKEYMAP
       io_Flags        IOF_QUICK if quick I/O possible, else zero
       io_Length       sizeof(*keyMap)
       io_Data         struct KeyMap *keyMap
                       pointer to a structure that describes
                       the raw keycode to byte stream conversion.

   RESULTS
       This function sets the io_Error field in the IOStdReq, and fills
       the current device default key map from the structure pointed to
       by io_Data.

   BUGS

   SEE ALSO
       exec/io.h, devices/keymap.h, devices/console.h
```

```
console.device/CD_SETKEYMAP                           console.device/CD_SETKEYMAP

    NAME
        CD_SETKEYMAP -- set the current key map structure for this console

    FUNCTION
        Set the current KeyMap structure used by this console unit to
        the structure pointed to by io_Data.

    IO REQUEST
        io_Message      mn_ReplyPort set if quick I/O is not possible
        io_Device       preset by the call to OpenDevice
        io_Unit         preset by the call to OpenDevice
        io_Command      CD_SETKEYMAP
        io_Flags        IOF_QUICK if quick I/O possible, else zero
        io_Length       sizeof(*keyMap)
        io_Data         struct KeyMap *keyMap
                        pointer to a structure that describes
                        the raw keycode to byte stream conversion.

    RESULTS
        This function sets the io_Error field in the IOStdReq, and fills
        the current key map from the structure pointed to by io_Data.

    BUGS

    SEE ALSO
        exec/io.h, devices/keymap.h, devices/console.h
```

```
console.device/CDInputHandler                       console.device/CDInputHandler

    NAME
        CDInputHandler -- handle an input event for the console device

    SYNOPSIS
        events = CDInputHandler(events, consoleDevice)
                                a0      a1

    FUNCTION
        Accept input events from the producer, which is usually the
        rom input.task.

    INPUTS
        events - a pointer to a list of input events.
        consoleDevice - a pointer to the library base address of the
            console device.  This has the same value as ConsoleDevice
            described below.

    RESULTS
        events - a pointer to a list of input events not used by this
            handler.

    NOTES
        This function is available for historical reasons.  It is
        preferred that input events be fed to the system via the
        WriteEvent command of the input.device.

        This function is different from standard device commands in
        that it is a function in the console device library vectors.
        In order to obtain a valid library base pointer for the
        console device (a.k.a. ConsoleDevice) call
        OpenDevice("console.device", -1, IOStdReq, 0),
        and then grab the io_Device pointer field out of the IOStdReq
        and use as ConsoleDevice.

    BUGS

    SEE ALSO
        input.device
```

console.device

```
console.device/CMD_CLEAR                               console.device/CMD_CLEAR

    NAME
        CMD_CLEAR -- Clear console input buffer.

    FUNCTION
        Remove from the console input buffer any reports waiting to
        satisfy read requests.

    IO REQUEST INPUT
        io_Message      mn_ReplyPort set if quick I/O is not possible
        io_Device       preset by the call to OpenDevice
        io_Unit         preset by the call to OpenDevice
        io_Command      CMD_CLEAR
        io_Flags        IOB_QUICK set if quick I/O is possible, else 0

    SEE ALSO
        exec/io.h, devices/console.h
```

console.device

```
console.device/CMD_READ                                 console.device/CMD_READ

   NAME
       CMD_READ -- return the next input from the keyboard

   FUNCTION
       Read the next input, generally from the keyboard.  The form of
       this input is as an ANSI byte stream: i.e. either ASCII text
       or control sequences.  Raw input events received by the
       console device can be selectively filtered via the aSRE and aRRE
       control sequences (see the write command).  Keys are converted
       via the keymap associated with the unit, which is modified
       with AskKeyMap and SetKeyMap

       If, for example, raw keycodes had been enabled by writing
       <CSI>1{ to the console (where <CSI> is $9B or Esc[), keys
       would return raw keycode reports with the information from
       the input event itself, in the form:
       <CSI>1;0;<keycode>;<qualifiers>;0;0;<seconds>;<microseconds>q

       If there is no pending input, this command will not be
       satisfied, but if there is some input, but not as much as can
       fill io_Length, the request will be satisfied with the input
       currently available.

    IO REQUEST
       io_Message      mn_ReplyPort set if quick I/O is not possible
       io_Device       preset by the call to OpenDevice
       io_Unit         preset by the call to OpenDevice
       io_Command      CMD_READ
       io_Flags        IOF_QUICK if quick I/O possible, else zero
       io_Length       sizeof(*buffer)
       io_Data         char buffer[]
                       a pointer to the destination for the characters to read
                       from the keyboard.

   RESULTS
       This function sets the error field in the IOStdReq, and fills
           in the io_Data area with the next input, and io_Actual with
           the number of bytes read.

    BUGS

    SEE ALSO
       exec/io.h, devices/console.h
```

```
console.device/CMD_WRITE                              console.device/CMD_WRITE

   NAME
        CMD_WRITE -- Write ANSI text to the console display.

   FUNCTION
        Write a text record to the display.  Interpret the ANSI
        control characters in the data as described below.  Note
        that the RPort of the console window is in use while this
        write command is pending.

   IO REQUEST INPUT
        io_Message      mn_ReplyPort set if quick I/O is not possible
        io_Device       preset by the call to OpenDevice
        io_Unit         preset by the call to OpenDevice
        io_Command      CMD_WRITE
        io_Flags        IOF_QUICK if quick I/O possible, else zero
        io_Length       sizeof(*buffer), or -1 if io_Data is null
                        terminated
        io_Data         a pointer to a buffer containing the ANSI text
                        to write to the console device.

   IO REQUEST RESULTS
        io_Error        the error result (no errors are reported as of V36)
        io_Actual       the number of bytes written from io_Data
        io_Length       zero
        io_Data         original io_Data plus io_Actual

   ANSI CODES SUPPORTED

        Codes are specified in the standard fashion for ANSI documents,
        as the two 4 bit nibbles that comprise the character code,
        high nibble first, separated by a slash.  Thus 01/11 (ESC) is
        a character with the hex value 1B (or the decimal value 27).

        A character on the Amiga falls into one of the following four
        ranges:
        00/ 0-01/15     C0: ASCII control characters.  See below.
        02/ 0-07/15     G0: ASCII graphic characters.  These characters
                        have an image that is displayed.  Note that the
                        DEL character is displayed by the Console Device:
                        it is not treated as control character here.
        08/ 0-09/15     C1: ANSI 3.41 control characters.  See below.
        10/ 0-15/15     G1: ECMA 94 Latin 1 graphic characters.

        Independent Control Functions (no introducer) --
        Code     Name    Definition
        -----    ---     ---------------------------------------------
        00/ 7    BEL     BELL: actually an Intuition DisplayBeep()
        00/ 8    BS      BACKSPACE
        00/ 9    HT      HORIZONTAL TAB
        00/10    LF      LINE FEED
        00/11    VT      VERTICAL TAB
        00/12    FF      FORM FEED
        00/13    CR      CARRIAGE RETURN
        00/14    SO      SHIFT OUT: causes all subsequent G0 (ASCII)
                         characters to be shifted to G1 (ECMA 94/1)
                         characters.
        00/15    SI      SHIFT IN: cancels the effect of SHIFT OUT.
        01/11    ESC     ESCAPE

        Code or Esc Name Definition
        -----    --- ---- ---------------------------------------------
        08/ 4    D   IND  INDEX: move the active position down one line.
        08/ 5    E   NEL  NEXT LINE
        08/ 8    H   HTS  HORIZONTAL TABULATION SET
```

```
        08/13    M   RI   REVERSE INDEX
        09/11    [   CSI  CONTROL SEQUENCE INTRODUCER: see next list

        ISO Compatible Escape Sequences (introduced by Esc) --
        Esc   Name Definition
        ----- ---- ---------------------------------------------------
        c     RIS  RESET TO INITIAL STATE: reset the console display.

        Control Sequences, with the number of indicated parameters.
        i.e. <CSI><parameters><control sequence letter(s)>.  Note the
        last entries consist of a space and a letter.  CSI is either
        9B or Esc[.  A minus after the number of parameters (#p)
        indicates less is valid.  Parameters are separated by
        semicolons, e.g. Esc[14;80H sets the cursor position to row
        14, column 80.
        CSI #p  Name Definition
        --- --- ---- -------------------------------------------------
        @   1-  ICH  INSERT CHARACTER
        A   1-  CUU  CURSOR UP
        B   1-  CUD  CURSOR DOWN
        C   1-  CUF  CURSOR FORWARD
        D   1-  CUB  CURSOR BACKWARD
        E   1-  CNL  CURSOR NEXT LINE
        F   1-  CPL  CURSOR PRECEDING LINE
        H   2-  CUP  CURSOR POSITION
        I   1-  CHT  CURSOR HORIZONTAL TABULATION
        J   1-  ED   ERASE IN DISPLAY (only to end of display)
        K   1-  EL   ERASE IN LINE (only to end of line)
        L   1-  IL   INSERT LINE
        M   1-  DL   DELETE LINE
        P   1-  DCH  DELETE CHARACTER
        R   2   CPR  CURSOR POSITION REPORT (in Read stream only)
        S   1-  SU   SCROLL UP
        T   1-  SD   SCROLL DOWN
        W   n   CTC  CURSOR TABULATION CONTROL
        Z   1-  CBT  CURSOR BACKWARD TABULATION
        f   2-  HVP  HORIZONTAL AND VERTICAL POSITION
        g   1-  TBC  TABULATION CLEAR
        h   n   SM   SET MODE: see modes below.
        l   n   RM   RESET MODE: see modes below.
        m   n   SGR  SELECT GRAPHIC RENDITION
        n   1-  DSR  DEVICE STATUS REPORT
        t   1-  aSLPP SET PAGE LENGTH (private Amiga sequence)
        u   1-  aSLL SET LINE LENGTH (private Amiga sequence)
        x   1-  aSLO SET LEFT OFFSET (private Amiga sequence)
        y   1-  aSTO SET TOP OFFSET (private Amiga sequence)
        {   n   aSRE SET RAW EVENTS (private Amiga sequence)
        |   8   aIER INPUT EVENT REPORT (private Amiga Read sequence)
        }   n   aRRE RESET RAW EVENTS (private Amiga sequence)
        ~   1   aSKR SPECIAL KEY REPORT (private Amiga Read sequence)
         p  1-  aSCR SET CURSOR RENDITION (private Amiga sequence)
         q  0   aWSR WINDOW STATUS REQUEST (private Amiga sequence)
         r  4   aWBR WINDOW BOUNDS REPORT (private Amiga Read sequence)
         v  1   aRAV RIGHT AMIGA V PRESS (private Amiga Read sequence-V37)

        Modes, set with <CSI><mode-list>h, and cleared with
        <CSI><mode-list>l, where the mode-list is one or more of the
        following parameters, separated by semicolons --
        Mode    Name Definition
        ------- ---- -------------------------------------------------
        20      LNM  LINEFEED NEWLINE MODE: if a linefeed is a newline
        >1      ASM  AUTO SCROLL MODE: if scroll at bottom of window
        ?7      AWM  AUTO WRAP MODE: if wrap at right edge of window

   NOTES
        The console.device recognizes these SGR sequences.
```

console.device

Page 11

```
    Note that some of these are new to V36.

    SGR (SELECT GRAPHICS RENDITION)
            Selects colors, and other display characteristics
            for text.

    Syntax:
            <ESC>[graphic-rendition...m

    Example:
            <ESC>[1;7m   (sets bold, and reversed text)

    Parameters:

            0       - Normal colors, and attributes
            1       - Set bold
            2       - Set faint (secondary color)
            3       - Set italic
            4       - Set underscore
            7       - Set reversed character/cell colors
            8       - Set concealed mode.
            22      - Set normal color, not bold    (V36)
            23      - Italic off                    (V36)
            24      - Underscore off                (V36)
            27      - Reversed off                  (V36)
            28      - Concealed off                 (V36)

            30-37   - Set character color
            39      - Reset to default character color

            40-47   - Set character cell color
            49      - Reset to default character cell color

            >0-7    - Set background color          (V36)
                      Used to set the background color before
                      any text is written.  The numeric parameter
                      is prefixed by ">".  This also means that if
                      you issue an SGR command with more than one
                      parameter, you must issue the digit only
                      parameters first, followed by any prefixed
                      parameters.

BUGS
    Does not correctly display cursor in SuperBitMap layers for
    versions prior to V36.

SEE ALSO
    ROM Kernal Manual (Volume 1), exec/io.h
```

console.device

Page 12

```
console.device/OpenDevice                               console.device/OpenDevice

   NAME
       OpenDevice -- a request to open a Console device

   SYNOPSIS
       error = OpenDevice("console.device", unit, IOStdReq, flags )
       d0                 a0                d0    a1        d1

   FUNCTION
       The open routine grants access to a device.  There are two
       fields in the IOStdReq block that will be filled in: the
       io_Device field and possibly the io_Unit field.

       As of (V37) the flags field may also be filled in with
       a value described below (see conunit.h or conunit.i).

       This open command differs from most other device open commands
       in that it requires some information to be supplied in the
       io_Data field of the IOStdReq block.  This initialization
       information supplies the window that is used by the console
       device for output.

       The unit number that is a standard parameter for an open call
       is used specially by this device.  See conunit.h, or conunit.i
       for defined valid unit numbers.


       unit number: -1 (CONU_LIBRARY)

               Used to get a pointer to the device library vector
       which is returned in the io_Device field of the IOStdReq
       block.  No actual console is opened.  You must still close
       the device when you are done with it.

       unit number: 0 (CONU_STANDARD)

               A unit number of zero binds the supplied window to
       a unique console.  Sharing a console must be done at a level
       higher than the device.


       unit number: 1 (CONU_CHARMAP) (V36)

               A unit number of one is similar to a unit number of
       zero, but a console map is also created, and maintained by
       the console.device.  The character map is used by the console
       device to restore obscured portions of windows which are
       revealed, and to redraw a window after a resize.  Character
       mapped console.device windows must be opened as SIMPLE REFRESH
       windows.

               The character map is currently for internal use
       only, and is not accessible by the programmer.  The character
       map stores characters, attributes, and style information for
       each character written with the CMD_WRITE command.

       unit number: 3 (CONU_SNIPMAP) (V36)

               A unit number of three is similar to a unit number
       of one, but also gives the user the ability to highlight
       text with the mouse which can be copied by pressing
       RIGHT AMIGA C.  See NOTES below.


       flags: 0 (CONFLAG_DEFAULT)
```

```
        The flags field should be set to 0 under V34, or less.

    flags: 1 (CONFLAG_NODRAW_ON_NEWSIZE) (V37)

        The flags field can be set to 0, or 1 as of V37.  The
    flags field is ignored under V36, so can be set, though it
    will have no effect.  When set to 1, it means that you don't
    want the console.device to redraw the window when the window
    size is changed (assuming you have opened the console.device
    with a character map - unit numbers 1, or 3).  This flag is
    ignored if you have opened a console.device with a unit
    number of 0.  Typically you would use this flag when you
    want to perform your own window refresh on a newsize, and
    you want the benefits of a character mapped console.

IO REQUEST
    io_Data         struct Window *window
                    This is the window that will be used for this
                    console.  It must be supplied if the unit in
                    the OpenDevice call is 0 (see above).  The
                    RPort of this window is potentially in use by
                    the console whenever there is an outstanding
                    write command.
INPUTS
    "console.device" - a pointer to the name of the device to be opened.
    unit - the unit number to open on that device.
    IOStdReq - a pointer to a standard request block
    0 - a flag field of zero (CONFLAG_DEFAULT)
    1 - a flag field of one  (CONFLAG_NODRAW_ON_NEWSIZE) (V37)

RESULTS
    error - zero if successful, else an error is returned.

NOTES
    As noted above, opening the console.device with a unit number of 3
    allows the user to drag select text, and copy the selection with
    RIGHT AMIGA C.  The snip is copied to a private buffered managed
    by the console.device (as of V36).  The snip can be copied to
    any console.device window unless you are running a console to
    clipboard utility such as that provided with V37.

    The user pastes text into console.device windows by pressing
    RIGHT AMIGA V.  Both RIGHT AMIGA V, and RIGHT AMIGA C are swallowed
    by the console.device (unless you have asked for key presses as
    RAW INPUT EVENTS).  Text pasted in this way appears in the
    console read stream as if the user had typed all of the characters
    manually.  Additional input (e.g., user input, RAW INPUT EVENTS)
    are queued up after pastes.  Pastes can theoretically be quite
    large, though they are no larger than the amount of text
    which is visible in a console.device window.

    When running the console to clipboard utility, text snips
    are copied to the clipboard.device, and RIGHT AMIGA V key
    presses are broadcast as an escape sequence as part of the
    console.device read stream ("<CSI>0 v" - $9B,$30,$20,$76).

    It is left up to the application to decide what to do when this
    escape sequence is received.  Ideally the application
    will read the contents of the clipboard, and paste the text
    by using successive writes to the console.device.

    Because the contents of the clipboard.device can be quite
    large, your program should limit the size of writes to something
    reasonable (e.g., no more than 1K characters per CMD_WRITE, and
    ideally no more than 256 characters per write).  Your program
```

```
    should continue to read events from the console.device looking
    for user input, and possibly RAW INPUT EVENTS.  How you decide
    to deal with these events is left up to the application.

    If you are using a character mapped console you should receive
    Intuition events as RAW INPUT EVENTS from the console.device.
    By doing this you will hear about these events after the console
    device does.  This allows the console.device to deal with events
    such as window resizing, and refresh before your application.

BUGS

SEE ALSO
    exec/io.h, intuition/intuition.h
```

```
console.device/RawKeyConvert                          console.device/RawKeyConvert

    NAME
        RawKeyConvert -- decode raw input classes

    SYNOPSIS
        actual = RawKeyConvert(event, buffer, length, keyMap)
        D0                      A0     A1      D1      A2

        ConsoleDevice in A6 if called from Assembly Language.

    FUNCTION
        This console function converts input events of type
        IECLASS_RAWKEY to ANSI bytes, based on the keyMap, and
        places the result into the buffer.

    INPUTS
        event -  an InputEvent structure pointer.
        buffer - a byte buffer large enough to hold all anticipated
            characters generated by this conversion.
        length - maximum anticipation, i.e. the buffer size in bytes.
        keyMap - a KeyMap structure pointer, or null if the default
            console device key map is to be used.

    RESULTS
        actual - the number of characters in the buffer, or -1 if
            a buffer overflow was about to occur.

    ERRORS
        if actual is -1, a buffer overflow condition was detected.
        Not all of the characters in the buffer are valid.

    NOTES
        This function is different from standard device commands in
        that it is a function in the console device library vectors.
        In order to obtain a valid library base pointer for the
        console device (a.k.a. ConsoleDevice) call
        OpenDevice("console.device", -1, IOStdReq, 0),
        and then grab the io_Device pointer field out of the IOStdReq
        and use as ConsoleDevice.

    BUGS

    SEE ALSO
        exec/io.h, devices/inputevent.h, devices/keymap.h
```

gameport.device Page 1

TABLE OF CONTENTS

gameport.device/GPD_ASKCTYPE
gameport.device/GPD_ASKTRIGGER
gameport.device/GPD_READEVENT
gameport.device/GPD_SETCTYPE
gameport.device/GPD_SETTRIGGER

---

gameport.device Page 2

```
gameport.device/GPD_ASKCTYPE                          gameport.device/GPD_ASKCTYPE

   NAME
        GPD_ASKCTYPE -- Acquire the current game port controller type

   FUNCTION
        This command identifies the type of controller at the game
        port, so that the signals at the port may be properly
        interpreted.  The controller type has been set by a previous
        SetCType.

        This command always executes immediately.

   IO REQUEST
        io_Message      mn_ReplyPort set if quick I/O is not possible
        io_Device       preset by the call to OpenDevice
        io_Unit         preset by the call to OpenDevice
        io_Command      GPD_ASKCTYPE
        io_Flags        IOB_QUICK set if quick I/O is possible
        io_Length       at least 1
        io_Data         the address of the byte variable for the
                        result
```

gameport.device Page 3

```
gameport.device/GPD_ASKTRIGGER               gameport.device/GPD_ASKTRIGGER

   NAME
        GPD_ASKTRIGGER -- Inquire the conditions for a game port report

   FUNCTION
        This command inquires what conditions must be met by a game
        port unit before a pending Read request will be satisfied.
        These conditions, called triggers, are independent -- that
        any one occurs is sufficient to queue a game port report to
        the Read queue.  These conditions are set by SetTrigger.

        This command always executes immediately.

   IO REQUEST
        io_Message      mn_ReplyPort set if quick I/O is not possible
        io_Device       preset by the call to OpenDevice
        io_Unit         preset by the call to OpenDevice
        io_Command      GPD_ASKTRIGGER
        io_Flags        IOB_QUICK set if quick I/O is possible
        io_Length       sizeof(gamePortTrigger)
        io_Data         a structure of type GamePortTrigger, which
                        has the following elements
            gpt_Keys -
                    GPTB_DOWNKEYS set if button down transitions
                    trigger a report, and GPTB_UPKEYS set if button up
                    transitions trigger a report
            gpt_Timeout -
                    a time which, if exceeded, triggers a report;
                    measured in vertical blank units (60/sec)
            gpt_XDelta  -
                    a distance in x which, if exceeded, triggers a
                    report
            gpt_YDelta  -
                    a distance in x which, if exceeded, triggers a
                    report
```

gameport.device Page 4

```
gameport.device/GPD_READEVENT                 gameport.device/GPD_READEVENT

   NAME
        GPD_READEVENT -- Return the next game port event.

   FUNCTION
        Read game port events from the game port and put them in the
        data area of the iORequest.  If there are no pending game port
        events, this command will not be satisfied, but if there are
        some events, but not as many as can fill IO_LENGTH, the
        request will be satisfied with those currently available.

   IO REQUEST
        io_Message      mn_ReplyPort set if quick I/O is not possible
        io_Device       preset by the call to OpenDevice
        io_Unit         preset by the call to OpenDevice
        io_Command      GPD_READEVENT
        io_Flags        IOB_QUICK set if quick I/O is possible
        io_Length       the size of the io_Data area in bytes: there
                        are sizeof(inputEvent) bytes per input event.
        io_Data         a buffer area to fill with input events.  The
                        fields of the input event are:
            ie_NextEvent
                        links the events returned
            ie_Class
                        is IECLASS_RAWMOUSE
            ie_SubClass
                        is 0 for the left, 1 for the right game port
            ie_Code
                        contains any gameport button reports.  No
                        report is indicated by the value 0xff.
            ie_Qualifier
                        only the relative and button bits are set
            ie_X, ie_Y
                        the x and y values for this report, in either
                        relative or absolute device dependent units.
            ie_TimeStamp
                        the delta time since the last report, given
                        not as a standard timestamp, but as the frame
                        count in the TV_SECS field.

   RESULTS
        This function sets the error field in the iORequest, and fills
        the iORequest with the next game port events (but not partial
        events).

   SEE ALSO
        gameport.device/SetCType, gameport.device/SetTrigger
```

gameport.device Page 5

```
gameport.device/GPD_SETCTYPE                    gameport.device/GPD_SETCTYPE

   NAME
        GPD_SETCTYPE -- Set the current game port controller type

   FUNCTION
        This command sets the type of device at the game port, so that
        the signals at the port may be properly interpreted.  The port
        can also be turned off, so that no reports are generated.

        This command always executes immediately.

   IO REQUEST
        io_Message      mn_ReplyPort set if quick I/O is not possible
        io_Device       preset by the call to OpenDevice
        io_Unit         preset by the call to OpenDevice
        io_Command      GPD_SETCTYPE
        io_Flags        IOB_QUICK set if quick I/O is possible
        io_Length       1
        io_Data         the address of the byte variable describing
                        the controller type, as per the equates in
                        the gameport include file
```

gameport.device Page 6

```
gameport.device/GPD_SETTRIGGER                gameport.device/GPD_SETTRIGGER

   NAME
        GPD_SETTRIGGER -- Set the conditions for a game port report

   FUNCTION
        This command sets what conditions must be met by a game
        port unit before a pending Read request will be satisfied.
        These conditions, called triggers, are independent -- that
        any one occurs is sufficient to queue a game port report to
        the Read queue.  These conditions are inquired with
        AskTrigger.

        This command always executes immediately.

   IO REQUEST
        io_Message      mn_ReplyPort set if quick I/O is not possible
        io_Device       preset by the call to OpenDevice
        io_Unit         preset by the call to OpenDevice
        io_Command      GPD_SETTRIGGER
        io_Flags        IOB_QUICK set if quick I/O is possible
        io_Length       sizeof(gamePortTrigger)
        io_Data         a structure of type GamePortTrigger, which
                        has the following elements
            gpt_Keys -
                    GPTB_DOWNKEYS set if button down transitions
                    trigger a report, and GPTB_UPKEYS set if button up
                    transitions trigger a report
            gpt_Timeout -
                    a time which, if exceeded, triggers a report;
                    measured in vertical blank units (60/sec)
            gpt_XDelta  -
                    a distance in x which, if exceeded, triggers a
                    report
            gpt_YDelta  -
                    a distance in x which, if exceeded, triggers a
                    report
```

input.device Page 1

```
TABLE OF CONTENTS

input.device/IND_ADDHANDLER
input.device/IND_REMHANDLER
input.device/IND_SETMPORT
input.device/IND_SETMTRIG
input.device/IND_SETMTYPE
input.device/IND_SETPERIOD
input.device/IND_SETTHRESH
input.device/IND_WRITEEVENT
input.device/PeekQualifier
```

input.device Page 2

```
input.device/IND_ADDHANDLER                         input.device/IND_ADDHANDLER

   NAME
        IND_ADDHANDLER -- Add an input handler to the device

   FUNCTION
        Add a function to the list of functions called to handle
        input events generated by this device.  The function is called
        as
            newInputEvents = Handler(inputEvents, handlerData);
            D0                        A0            A1

   IO REQUEST
        io_Message      mn_ReplyPort set
        io_Device       preset by OpenDevice
        io_Unit         preset by OpenDevice
        io_Command      IND_ADDHANDLER
        io_Data         a pointer to an interrupt structure.
            is_Data     the handlerData pointer described above
            is_Code     the Handler function address

   NOTES
        The interrupt structure is kept by the input device until a
        RemHandler command is satisfied for it.
```

```
input.device/IND_REMHANDLER                      input.device/IND_REMHANDLER

   NAME
        IND_REMHANDLER -- Remove an input handler from the device

   FUNCTION
        Remove a function previously added to the list of handler
        functions.

   IO REQUEST
        io_Message      mn_ReplyPort set
        io_Device       preset by OpenDevice
        io_Unit         preset by OpenDevice
        io_Command      IND_REMHANDLER
        io_Data         a pointer to the interrupt structure.

   NOTES
        This command is not immediate
```

```
input.device/IND_SETMPORT                             input.device/IND_SETMPORT

   NAME
        IND_SETMPORT -- Set the current mouse port

   FUNCTION
        This command sets the gameport port at which the mouse is
        connected.

   IO REQUEST
        io_Message      mn_ReplyPort set if quick I/O is not possible
        io_Device       preset by the call to OpenDevice
        io_Unit         preset by the call to OpenDevice
        io_Command      IND_SETMPORT
        io_Flags        IOB_QUICK set if quick I/O is possible
        io_Length       1
        io_Data         a pointer to a byte that is either 0 or 1,
                        indicating that mouse input should be obtained
                        from either the left or right controller port,
                        respectively.
```

```
input.device/IND_SETMTRIG                               input.device/IND_SETMTRIG

   NAME
        IND_SETMTRIG -- Set the conditions for a mouse port report

   FUNCTION
        This command sets what conditions must be met by a mouse
        before a pending Read request will be satisfied.  The trigger
        specification is that used by the gameport device.

   IO REQUEST
        io_Message      mn_ReplyPort set if quick I/O is not possible
        io_Device       preset by the call to OpenDevice
        io_Unit         preset by the call to OpenDevice
        io_Command      IND_SETMTRIG
        io_Flags        IOB_QUICK set if quick I/O is possible
        io_Length       sizeof(gameportTrigger)
        io_Data         a structure of type GameportTrigger, which
                        has the following elements
            gpt_Keys -
                    GPTB_DOWNKEYS set if button down transitions
                    trigger a report, and GPTB_UPKEYS set if button up
                    transitions trigger a report
            gpt_Timeout -
                    a time which, if exceeded, triggers a report;
                    measured in vertical blank units (60/sec)
            gpt_XDelta  -
                    a distance in x which, if exceeded, triggers a
                    report
            gpt_YDelta  -
                    a distance in x which, if exceeded, triggers a
                    report
```

```
input.device/IND_SETMTYPE                               input.device/IND_SETMTYPE

   NAME
        IND_SETMTYPE -- Set the current mouse port controller type

   FUNCTION
        This command sets the type of device at the mouse port, so
        the signals at the port may be properly interpreted.

   IO REQUEST
        io_Message      mn_ReplyPort set if quick I/O is not possible
        io_Device       preset by the call to OpenDevice
        io_Unit         preset by the call to OpenDevice
        io_Command      IND_SETMTYPE
        io_Flags        IOB_QUICK set if quick I/O is possible
        io_Length       1
        io_Data         the address of the byte variable describing
                        the controller type, as per the equates in
                        the gameport include file
```

```
input.device/IND_SETPERIOD                          input.device/IND_SETPERIOD

   NAME
        IND_SETPERIOD -- Set the key repeat period

   FUNCTION
        This command sets the period at which a repeating key repeats.

        This command always executes immediately.

   IO REQUEST - a timerequest
        tr_node.io_Message      mn_ReplyPort set if quick I/O is not possible
        tr_node.io_Device       preset by the call to OpenDevice
        tr_node.io_Unit         preset by the call to OpenDevice
        tr_node.io_Command      IND_SETPERIOD
        tr_node.io_Flags        IOB_QUICK set if quick I/O is possible
        tr_time.tv_secs         the repeat period seconds
        tr_time.tv_micro        the repeat period microseconds
```

```
input.device/IND_SETTHRESH                          input.device/IND_SETTHRESH

   NAME
        IND_SETTHRESH -- Set the key repeat threshold

   FUNCTION
        This command sets the time that a key must be held down before
        it can repeat.  The repeatability of a key may be restricted
        (as, for example, are the shift keys).

        This command always executes immediately.

   IO REQUEST - a timerequest
        tr_node.io_Message      mn_ReplyPort set if quick I/O is not possible
        tr_node.io_Device       preset by the call to OpenDevice
        tr_node.io_Unit         preset by the call to OpenDevice
        tr_node.io_Command      IND_SETTHRESH
        tr_node.io_Flags        IOB_QUICK set if quick I/O is possible
        tr_time.tv_secs         the threshold seconds
        tr_time.tv_micro        the threshold microseconds
```

input.device

Page 9

```
input.device/IND_WRITEEVENT                    input.device/IND_WRITEEVENT

   NAME
       IND_WRITEEVENT -- Propagate an input event to all handlers

   FUNCTION

   IO REQUEST
       io_Message      mn_ReplyPort set if quick I/O is not possible
       io_Device       preset by the call to OpenDevice
       io_Unit         preset by the call to OpenDevice
       io_Command      IND_WRITEEVENT
       io_Flags        IOB_QUICK set if quick I/O is possible
       io_Length       should be sizeof(struct InputEvent)
       io_Data         a pointer to the struct InputEvent:
           ie_NextEvent
                       will be ignored.
           ie_Class
           ie_SubClass
           ie_Code
           ie_Qualifier
           ie_X, ie_Y
                       as desired
           ie_TimeStamp
                       will be set by this call (V36)

   NOTES
       The contents of the input event are destroyed.

       This function was documented in V34 and earlier to allow
       chaining of events via ie_NextEvent.  The implementation
       never allowed that.  The documentation now reflects this.

       ie_TimeStamp is set only in V36 and later.  Software written
       to run on earlier versions should set this field to the
       current time.
```

input.device

Page 10

```
input.device/PeekQualifier                        input.device/PeekQualifier

   NAME
       PeekQualifier -- get the input device's current qualifiers (V36)

   SYNOPSIS
       qualifier = PeekQualifier()
       d0

       UWORD PeekQualifier();

   FUNCTION
       This function takes a snapshot of what the input device thinks
       the current qualifiers are.

   RESULTS
       qualifier - a word with the following bits set according to
           what the input device knows their state to be:
               IEQUALIFIER_LSHIFT, IEQUALIFIER_RSHIFT,
               IEQUALIFIER_CAPSLOCK, IEQUALIFIER_CONTROL,
               IEQUALIFIER_LALT, IEQUALIFIER_RALT,
               IEQUALIFIER_LCOMMAND, IEQUALIFIER_RCOMMAND,
               IEQUALIFIER_LEFTBUTTON, IEQUALIFIER_RBUTTON,
               IEQUALIFIER_MIDBUTTON

   NOTE
       This function is new for V36.

   SEE ALSO
       devices/inputevent.h
```

TABLE OF CONTENTS

keyboard.device/CMD_CLEAR
keyboard.device/KBD_ADDRESETHANDLER
keyboard.device/KBD_READEVENT
keyboard.device/KBD_READMATRIX
keyboard.device/KBD_REMRESETHANDLER
keyboard.device/KBD_RESETHANDLERDONE

```
keyboard.device/CMD_CLEAR                           keyboard.device/CMD_CLEAR

   NAME
        CMD_CLEAR -- Clear the keyboard input buffer.

   FUNCTION
        Remove from the input buffer any keys transitions waiting to
        satisfy read requests.

   IO REQUEST
        io_Message      mn_ReplyPort set if quick I/O is not possible
        io_Device       preset by the call to OpenDevice
        io_Command      CMD_CLEAR
        io_Flags        IOB_QUICK set if quick I/O is possible
```

keyboard.device

Page 3

```
keyboard.device/KBD_ADDRESETHANDLER          keyboard.device/KBD_ADDRESETHANDLER

   NAME
        KBD_ADDRESETHANDLER -- Add a keyboard reset handler.

   FUNCTION
        Add a function to the list of functions called to clean up
        before a hard reset generated at the keyboard.  The reset
        handler is called as:
            ResetHandler(handlerData)
                         a1

   IO REQUEST
        io_Message      mn_ReplyPort set
        io_Device       preset by OpenDevice
        io_Unit         preset by OpenDevice
        io_Command      KBD_ADDRESETHANDLER
        io_Data         a pointer to an interrupt structure.
            is_Data     the handlerData pointer described above
            is_Code     the Handler function address

   NOTES
        Few of the Amiga keyboard models generate the communication codes
        used to implement this reset processing.  Specifically, only the
        Euro a1000 (rare), and the B2000 keyboard generate them.

        The interrupt structure is kept by the keyboard device until a
        RemResetHandler command is satisfied for it, but the
        KBD_ADDRESETHANDLER command itself is replied immediately.
```

---

keyboard.device

Page 4

```
keyboard.device/KBD_READEVENT                    keyboard.device/KBD_READEVENT

   NAME
        KBD_READEVENT -- Return the next keyboard event.

   FUNCTION
        Read raw keyboard events from the keyboard and put them in the
        data area of the iORequest.  If there are no pending keyboard
        events, this command will not be satisfied, but if there are
        some events, but not as many as can fill IO_LENGTH, the
        request will be satisfied with those currently available.

   IO REQUEST
        io_Message      mn_ReplyPort set if quick I/O is not possible
        io_Device       preset by the call to OpenDevice
        io_Command      KBD_READEVENT
        io_Flags        IOB_QUICK set if quick I/O is possible
        io_Length       the size of the io_Data area in bytes: there
                        are sizeof(inputEvent) bytes per input event.
        io_Data         a buffer area to fill with input events.  The
                        fields of the input event are:
            ie_NextEvent
                        links the events returned
            ie_Class
                        is IECLASS_RAWKEY
            ie_Code
                        contains the next key up/down reports
            ie_Qualifier
                        only the shift and numeric pad bits are set
            ie_SubClass, ie_X, ie_Y, ie_TimeStamp
                        are not used, and set to zero

   RESULTS
        This function sets the error field in the IORequest, and fills
        the IORequest with the next keyboard events (but not partial
        events).
```

```
keyboard.device/KBD_READMATRIX                    keyboard.device/KBD_READMATRIX

   NAME
        KBD_READMATRIX -- Read the current keyboard key matrix.

   FUNCTION
        This function reads the up/down state of every key in the
        key matrix.

   IO REQUEST INPUT
        io_Message      mn_ReplyPort set if quick I/O is not possible
        io_Device       preset by the call to OpenDevice
        io_Command      KBD_READMATRIX
        io_Flags        IOB_QUICK set if quick I/O is possible
        io_Length       the size of the io_Data area in bytes: this
                        must be big enough to hold the key matrix.
        io_Data         a buffer area to fill with the key matrix:
                        an array of bytes whose component bits reflect
                        each keys state: the state of the key for
                        keycode n is at bit (n MOD 8) in byte
                        (n DIV 8) of this matrix.

   IO REQUEST OUTPUT
        io_Error
             IOERR_BADLENGTH - the io_Length was not exactly 13 bytes.
                        The buffer is unchanged.  This is only returned
                        by V33/V34 kickstart.
        io_Actual       the number of bytes filled in io_Data with key
                        matrix data, i.e. the minimum of the supplied
                        length and the internal key matrix size.

   NOTE
        For V33/V34 Kickstart, io_Length must be set to exactly 13 bytes.

   RESULTS
        This function sets the error field in the IORequest, and sets
        matrix to the current key matrix.
```

```
keyboard.device/KBD_REMRESETHANDLER          keyboard.device/KBD_REMRESETHANDLER

   NAME
        KBD_REMRESETHANDLER -- Remove a keyboard reset handler.

   FUNCTION
        Remove a function previously added to the list of reset
        handler functions with KBD_ADDRESETHANDLER.

   IO REQUEST
        io_Message      mn_ReplyPort set
        io_Device       preset by OpenDevice
        io_Unit         preset by OpenDevice
        io_Command      KBD_REMRESETHANDLER
        io_Data         a pointer to the handler interrupt structure.
```

keyboard.device

Page 7

```
keyboard.device/KBD_RESETHANDLERDONE     keyboard.device/KBD_RESETHANDLERDONE

   NAME
        KBD_RESETHANDLERDONE -- Indicate that reset handling is done.

   FUNCTION
        Indicate that reset cleanup associated with the handler has
        completed.  This command should be issued by all keyboard
        reset handlers so that the reset may proceed.

   IO REQUEST
        io_Message      mn_ReplyPort set
        io_Device       preset by OpenDevice
        io_Unit         preset by OpenDevice
        io_Command      KBD_RESETHANDLERDONE
        io_Data         a pointer to the handler interrupt structure.

   NOTES
        The keyboard processor itself performs the hardware reset, and
        will time out and perform the reset even if some reset handlers
        have not indicated yet that the reset may proceed.  This
        timeout is several seconds.
```

TABLE OF CONTENTS

narrator.device/AbortIO
narrator.device/OpenDevice
narrator.device/CMD_FLUSH
narrator.device/CMD_READ
narrator.device/CMD_RESET
narrator.device/CMD_START
narrator.device/CMD_STOP
narrator.device/CMD_WRITE
narrator.device/CloseDevice

```
narrator.device/AbortIO                                narrator.device/AbortIO

   NAME
        AbortIO - Abort an IO request


   SYNOPSIS
        AbortIO(IORequest)
                   A1


   FUNCTION
        Exec library call to abort a specified READ or WRITE request.
        The IORequest may be in the queue or currently active.  If
        currently active, the request is immediately stopped and then
        removed.


   INPUTS
        Pointer to the IORequest block to be aborted.


   RESULTS
        io_Error field in the IORequest block set to #IOERR_ABORTED.


   SEE ALSO
```

narrator.device

Page 3

```
narrator.device/CloseDevice                        narrator.device/CloseDevice

   NAME
        CloseDevice - terminates access to the narrator device


   SYNOPSIS
        CloseDevice(IORequest)
                       A1

   FUNCTION
        Close invalidates the IO_UNIT and IO_DEVICE fields in the
        IORequest block, preventing subsequent IO until another
        OpenDevice.  CloseDevice also reduces the open count.  If
        the count goes to 0 and the expunge bit is set, the device
        is expunged.  If the open count goes to zero and the delayed
        expunge bit is not set, CloseDevice sets the expunge bit.


   INPUTS
        A valid IORequest block with its io_Message structure, and
        io_Device and io_Unit fields properly initialized.  These
        fields are initialized by OpenDevice.


   RESULTS
        CloseDevice invalidates the unit and device pointers in the
        IORequest block.


   SEE ALSO
```

narrator.device

Page 4

```
narrator.device/CMD_FLUSH                              narrator.device/CMD_FLUSH

   NAME
        CMD_FLUSH - Aborts all inprogress and queued requests


   SYNOPSIS
        Standard device command.


   FUNCTION
        Aborts all inprogress and queued speech requests.


   INPUTS
        Valid IORequest block with the io_Command field set to CMD_FLUSH.
        A valid IORequest block is one with its io_Message structure, and
        io_Device and io_Unit fields properly initialized.  The easiest
        way to insure proper initialization is to make a copy of the
        IORequest block after a successful OpenDevice call.


   RESULTS
        io_Error in IORequest block set to 0


   SEE ALSO
        Exec input/output documentation.
```

```
narrator.device/CMD_Read                               narrator.device/CMD_Read

   NAME
        CMD_READ - Query the narrator device for mouth shape or other
                   synchronization events.

   SYNOPSIS
        Standard device command.

   FUNCTION
        Currently, there are three events which the user can inquire
        about from the narrator device.  These are: mouth shape changes,
        start of word, and start of syllable.  Each read request returns
        information about any or all of these events as determined by
        the bits set in the sync field of the read IORequest block.  In
        the case of mouth shape changes, each shape returned is guaranteed
        to be different from the previously returned shape to allow
        updating to be done only when necessary.  Each read request is
        associated with a write request by information contained in the
        IORequest block used to open the device.  Since the first field
        in the read IORequest block is a write IORequest structure, this
        association is easily made by copying the write IORequest block
        (after the OpenDevice call) into the voice field of the read
        IORequest block.  If there is no write in progress or in the
        device input queue with the same pseudo unit number as the read
        request, the read will be returned to the user with an error.  This
        is also how the user knows that the write request has finished and
        that s/he should not issue any more reads.  Note that in this case
        the mouth shapes may not be different from previously returned values.

   INPUTS
        mouth_rb IORequest block with the voice field (a narrator_rb
        structure) copied from the associated write request with the
        following fields modified:

           io_Message - Pointer to message port for read request
           io_Command - CMD_READ
           io_Error   - Clear before issuing first read
           width      - 0
           height     - 0

   RESULTS
        As long as the speech is in progress, each read returns the
        following information in the mouth_rb IORequest block.

        If mouth shape changes are requested the following fields are
        modified:
           width  - Contains mouth width value in arbitrary units
           height - Contains mouth height value in arbitrary units
           shape  - Compressed form of mouth shapes (internal use only)

        ******  NEW FOR V37 NARRATOR

        If word synchronization is requested:
           sync   - Bit NDB_WORDSYNC is set

        If syllable synchronization is requested:
           sync   - Bit NDB_SYLSYNC is set

        Note that any or all of the above fields can be set and it is
        the user's responsibility to check for all possibilities.

   SEE ALSO
        CMD_WRITE
        Exec input/output documentation.
```

```
narrator.device/CMD_RESET                              narrator.device/CMD_RESET

   NAME
        CMD_RESET - Reset the device to a known state


   SYNOPSIS
        Standard device command.


   FUNCTION
        Resets the device as though it has just be initialized.
        Aborts all read/write requests whether active of enqueued.
        Restarts device if it has been stopped.


   INPUTS
        Valid IORequest block with the io_Command field set to CMD_RESET.
        A valid IORequest block is one with its io_Message structure, and
        io_Device and io_Unit fields properly initialized.  The easiest
        way to insure proper initialization is to make a copy of the
        IORequest block after a successful OpenDevice call.


   RESULTS


   SEE ALSO
        Exec input/output documentation.
```

narrator.device Page 7

```
narrator.device/CMD_START                              narrator.device/CMD_START

   NAME
        CMD_START - Restarts the device after a CMD_STOP command


   SYNOPSIS
        Standard device command.


   FUNCTION
        CMD_START restarts the currently active speech (if any) and
        allows queued requests to start.


   INPUTS
        Valid IORequest block with the io_Command field set to CMD_START
        A valid IORequest block is one with its io_Message structure, and
        io_Device and io_Unit fields properly initialized.  The easiest
        way to insure proper initialization is to make a copy of the
        IORequest block after a successful OpenDevice call.


   RESULTS
        io_Error set to 0.


   SEE ALSO
        Exec input/output documentation.
```

narrator.device Page 8

```
narrator.device/CMD_STOP                               narrator.device/CMD_STOP

   NAME
        CMD_STOP  - Stops the device.


   SYNOPSIS
        Standard device command.


   FUNCTION
        CMD_STOP halts the currently active speech (if any) and prevents
        any queued requests from starting.


   INPUTS
        Valid IORequest block with the io_Command field set to CMD_STOP.
        A valid IORequest block is one with its io_Message structure, and
        io_Device and io_Unit fields properly initialized.  The easiest
        way to insure proper initialization is to make a copy of the
        IORequest block after a successful OpenDevice call.


   RESULTS
        io_Error set to 0.


   SEE ALSO
        Exec input/output documentation.
```

```
narrator.device/CMD_WRITE                        narrator.device/CMD_WRITE

   NAME
       CMD_WRITE - Send speech request to the narrator device


   SYNOPSIS
       Standard device command.

   FUNCTION
       Sends a phonetic string to the narrator device to be spoken
       and, optionally, is used to direct the narrator device to
       return mouth shape changes, and word and syllable sync events
       in response to read requests from the user.  The phonetic string
       consists of ASCII characters representing the individual phonemes.
       Refer to the narrator device chapter of the libraries and devices
       volume of the ROM Kernel Manual for detailed information.


   INPUTS
       User IORequest block (struct narrator_rb as defined in .h file).
       The OpenDevice call will initialize the IORequest block to a
       "standard male" voice.  If you want to change any parms, do so
       after the OpenDevice call and before the DoIO (or SendIO/WaitIO).
       For a complete description of the narrator_rb structure, see the
       narrator.h or .i include file.  Note that the OpenDevice call does
       not initialize all the fields needed by the narrator device.  The
       IORequest fields which must be set by the user before issuing the
       write request are:

          io_Command - Set to CMD_WRITE

          io_Data    - Pointer to phonetic string

          io_Length  - Length of phonetic string

          ch_masks   - Array of audio channel selection masks (see audio
                       device documentation for description of this field)

          nm_masks   - Number of audio channel selection masks


          ****** NEW FOR V37 NARRATOR

          flags        - The bit NDB_NEWIORB must be set in the flags
                         field if any of the new features of the V37
                         narrator are used.  In additon, two new bits,
                         NDB_WORDSYNC and NDB_SYLSYNC are used to
                         specify that word and/or syllable sync events
                         are to be generated.

          F0enthusiasm - F0 excursion fator.  This controls the degree
                         of pitch movement for accented syllables.

          F0perturb    - F0 perturbation.  Controls degree of pitch
                         "shakeyness" across the utterance.

          F1adj        - F1, F2, and F3 pitch adjustments.  These are
          F2adj          M-15% step adjustments added to the formant
          F3adj          frequencies and can be used to create unique
                         sounding vocal effects.

          A1adj        - A1, A2, and A3 loudness adjustments.  These are
          A2adj          decibel adjustemts to the individual formant
```

```
          A3adj          loudnesses and can be used to create unique
                         sounding vocal effects.

          articulate   - Transition time modification.  Controls the
                         abruptness of the transitions between phonemes.

          centralize   - Degree of vowel centralization.  All vowels in an
                         utterance can be made to sound somewhat like a user
                         specified vowel, giving a sense of regionality to
                         the voice.  "centralize" specifies the degree of
                         movement (in percent) from the vowel in the input
                         utterance to a user specified vowel (see centphon).

          centphon     - Pointer to an ASCII string specifying the particular
                         vowel that centralize interpolates towards.  The
                         allowable vowel strings are:
                              "IY", "IH", "EH", "AE", "AA", "AH",
                              "AO", "OW", "UH", "ER", and "UW".

          AVbias       - Adjusts the voicing amplitude (in decibels).

          AFbias       - Adjusts the fricative amplitude (in decibels).

          priority     - User specified priority.  The synthesizer needs to
                         run as a high priority task while it is actually
                         speaking to prevent clicking and gurgling.  The
                         default priority is 100.  Users can set this field
                         to specify their own speaking priority.  CAUTION:
                         too high a value may lock out other necessary taks.

       In addition to producing synthetic speech, the narrator device
       also provides features for synchronizing the speech to animation
       or other user defined events.  There are three types of events
       that the user can request.  They are mouth shape changes, start of
       new word, and start of new syllable.  Mouth shape changes are
       requested by setting the mouths field of the IORequest block to a
       non-zero value.  Word and syllable sync events are requested by
       setting the NDB_WORDSYNC and/or NDB_SYLSYNC bits in the flags field
       of the IORequest block.  Note that word and syllable sync only work
       in V37 and later versions of the narrator device.


   RESULTS
       The narrator device range checks and performs other validity
       checks for all input parms.  If any input is in error, the device
       sets the io_Error field of the IORequest block to an appropriate
       value (see include files for error codes).  If everything is in
       order, the narrator device will produce the speech and clear the
       io_Error field.  The io_Actual field is set to the length of the
       input string that was actually processed.  If the return code
       indicates a phoneme error (ND_PhonErr), io_Actual is the NEGATIVE
       of the position in the input string where the error occured.


   SEE ALSO
       Read command.
       Audio device documentation.
       Exec input/output documentation.
```

```
narrator.device/OpenDevice                              narrator.device/OpenDevice

   NAME
       OpenDevice - opens the narrator device.


   SYNOPSIS
       error = OpenDevice("narrator.device",  unit, IORequest, flags);
        D0                       A0            D0     A1         D1

   FUNCTION
       The OpenDevice routine grants access to the narrator device.
       OpenDevice checks the unit number, and if non-zero, returns an
       error (ND_UnitErr).  If this is the first time the driver has
       been opened, OpenDevice will attempt to open the audio device
       and allocate the driver's static buffers.  If either of these
       operations fail, an error is returned.  See the .h and .i
       include files for possible error return codes.  Next, OpenDevice
       (done for all opens, not just the first one) initializes various
       fields in the user's IORequest block (see below).  If users wish
       to use non-default values for these parms, the values must be set
       after the open is done.  OpenDevice also assigns a pseudo unit
       number to the IORB for use in synchronizing read and write requests.
       See the read command for more details.  Finally, OpenDevice stores
       the device node pointer in the IORequest block and clears the
       delayed expunge bit.

       ***** NEW FOR V37 NARRATOR *****

       Several new fields in the IORequest block have been added for V37
       narrator.  These fields are initialized when the device is opened
       if the NDB_NEWIORB bit is set in the flags field of the user's
       IORequest block.  Note that NDB_NEWIORB is set in the IORequest
       block, NOT in the "flags" input parm to the OpenDevice call.


   INPUTS
       device      - "narrator.device"
       unit        - 0
       IORequest   - Pointer to the user's IORequest block
       flags       - 0


   RESULTS
       The narrator device will initialize the IORequest block as follows
       (assume IORB points to the IOrequest block):

       IORB->rate = 150;              /* Speaking rate in words/minute */
       IORB->pitch = 110;             /* Baseline pitch in Hertz      */
       IORB->mode = NATURALF0;        /* Pitch (F0) mode              */
       IORB->sex = MALE;              /* Sex of voice                 */
       IORB->volume = 64              /* Volume, full on              */
       IORB->sampfreq = 22200         /* Audio sampling freq          */
       IORB->mouths = 0               /* Don't generate sync events   */

       and if the NDB_NEWIORB bit is set:

       IORB->F0enthusiasm = 0         /* F0 excursion factor          */
       IORB->F0perturb = 32           /* F0 perturbation (in 32nds)   */
       IORB->F1adj = 0                /* F1 adjustment in M-15% steps */
       IORB->F2adj = 0                /* F2 adjustment in M-15% steps */
       IORB->F3adj = 0                /* F3 adjustment in M-15% steps */
       IORB->A1adj = 0                /* A1 adjustment in decibels    */
       IORB->A2adj = 0                /* A2 adjustment in decibels    */
```

```
       IORB->A3adj = 0                /* A3 adjustment in decibels    */
       IORB->articulate = 100         /* Transition time multiplier   */
       IORB->centralize = 0           /* Degree of vowel centralization */
       IORB->centphon = ""            /* Pointer to central ASCII phon  */
       IORB->AVbias = 0               /* AV bias                      */
       IORB->AFbias = 0               /* AF bias                      */
       IORB->priority = 100           /* Priority while speaking      */


   SEE ALSO
       The include files contain the complete IORequest block definition,
           default settings, and error return codes.
       Exec input/output documentation.
```

TABLE OF CONTENTS

parallel.device/CMD_CLEAR
parallel.device/CMD_FLUSH
parallel.device/CMD_READ
parallel.device/CMD_RESET
parallel.device/CMD_START
parallel.device/CMD_STOP
parallel.device/CMD_WRITE
parallel.device/OpenDevice
parallel.device/PDCMD_QUERY
parallel.device/PDCMD_SETPARAMS

```
parallel.device/CMD_CLEAR                          parallel.device/CMD_CLEAR

   NAME
        Clear -- clear the parallel port buffer

   FUNCTION
        This command just RTS's (no buffer to clear)

   IO REQUEST
        io_Message      mn_ReplyPort initialized
        io_Device       set by OpenDevice
        io_Unit         set by OpenDevice
        io_Command      CMD_CLEAR (05)
```

```
parallel.device/CMD_FLUSH                          parallel.device/CMD_FLUSH

   NAME
        Flush -- clear all queued I/O requests for the parallel port

   FUNCTION
        This command purges the read and write request queues for the
        parallel device. The currently active request is not purged.

   IO REQUEST
        io_Message      mn_ReplyPort initialized
        io_Device       set by OpenDevice
        io_Unit         set by OpenDevice
        io_Command      CMD_FLUSH (08)
```

```
parallel.device/CMD_READ                                    parallel.device/CMD_READ

   NAME
        Read -- read input from parallel port

   FUNCTION
        This command causes a stream of characters to be read from the
        parallel I/O register. The number of characters is specified in
        io_Length. The EOF and EOL modes are supported, but be warned that
        using these modes can result in a buffer overflow if the proper
        EOL or EOF character is not received in time. These modes should
        be used only when the sender and receiver have been designed to
        cooperate. A safety guard can be implemented to EOF by setting
        io_Length to a maximum allowed value. That cannot be done with EOL
        since the EOL mode is identified by io_Length=-1.

        The parallel.device has no internal buffer; if no read request has
        been made, pending input (i.e. handshake request) is not
        acknowledged.

   IO REQUEST
        io_Message      mn_ReplyPort initialized
        io_Device       set by OpenDevice
        io_Unit         set by OpenDevice
        io_Command      CMD_READ (02)
        io_Flags        If IOF_QUICK is set, driver will attempt Quick IO
        io_Length       number of characters to receive.
        io_Data         pointer where to put the data.

   RESULTS
        io_Error -- if the Read succeded, then io_Error will be null.
            If the Read failed, then io_Error will contain an error code.

   SEE ALSO
        parallel.device/PDCMD_SETPARAMS
```

parallel.device Page 5

```
parallel.device/CMD_RESET                               parallel.device/CMD_RESET

   NAME
        Reset -- reinitializes the parallel device

   FUNCTION
        This command resets the parallel device to its freshly initialized
        condition. It aborts all I/O requests both queued and current and
        sets the devices's flags and parameters to their boot-up time
        default values. At boot-up time the PTermArray is random, and it
        will be so also here.

   IO REQUEST
        io_Message      mn_ReplyPort initialized
        io_Device       set by OpenDevice
        io_Unit         set by OpenDevice
        io_Command      CMD_RESET (01)

   RESULTS
        Error -- if the Reset succeded, then io_Error will be null.
                 if the Reset failed, then the io_Error will be non-zero.
```

parallel.device Page 6

```
parallel.device/CMD_START                                  parallel.device/CMD_START

   NAME
        Start -- restart paused I/O over the parallel port

   FUNCTION
        This command restarts the current I/O activity on the parallel
        port by reactivating the handshaking sequence.

   IO REQUEST
        io_Message      mn_ReplyPort initialized
        io_Device       set by OpenDevice
        io_Unit         set by OpenDevice
        io_Command      CMD_START (07)

   SEE ALSO
        parallel.device/CMD_STOP
```

parallel.device

Page 7

```
parallel.device/CMD_STOP                              parallel.device/CMD_STOP

   NAME
        Stop -- pause current activity on the parallel device

   FUNCTION
        This command halts the current I/O activity on the parallel
        device by discontinuing the handshaking sequence. The stop and
        start commands may not be nested.

   IO REQUEST
        io_Message      mn_ReplyPort initialized
        io_Device       set by OpenDevice
        io_Unit         set by OpenDevice
        io_Command      CMD_STOP (06)

   SEE ALSO
        parallel.device/CMD_START
```

parallel.device

Page 8

```
parallel.device/CMD_WRITE                             parallel.device/CMD_WRITE

   NAME
        Write -- send output to parallel port

   FUNCTION
        This command causes a stream of characters to be written to the
        parallel output register. The number of characters is specified in
        io_Length, unless -1 is used, in which case output is sent until
        a zero byte occurs in the data. This is independent of, and may be
        used simultaneously with setting the EOFMODE in io_ParFlags and using
        the PTermArray to terminate the read or write.

   IO REQUEST
        io_Message      mn_ReplyPort initialized
        io_Device       set by OpenDevice
        io_Unit         set by OpenDevice
        io_Command      CMD_WRITE (03)
        io_Flags        If IOF_QUICK is set, driver will attempt Quick IO
        io_Length       number of characters to transmit, or if set
                        to -1 send until zero byte encountered
        io_Data         pointer to block of data to transmit

   RESULTS
        io_Error -- If the Write succeded, then io_Error will be null.
            If the Write failed, then io_Error will contain an error code.

   SEE ALSO
        parallel.device/PDCMD_SETPARAMS
parallel.device/OpenDevice                          parallel.device/OpenDevice

   NAME
        Open -- a request to open the parallel port

   SYNOPSIS
        error = OpenDevice("parallel.device", unit, ioExtPar, flags)
        D0                  A0                 D0    A1         D1

   FUNCTION
        This function allows the requestor software access to the parallel
        device.  Unless the shared-access bit (bit 5 of io_ParFlags) is
        set, exclusive use is granted and no other access is allowed
        until the owner closes the device.

        A FAST_MODE, can be specified (bit 3 of io_Parflags) to speed up
        transfers to high-speed printers. Rather than waiting for the printer
        to acknowledge a character using the *ACK interrupt, this mode will
        send out data as long as the BUSY signal is low. The printer must be
        able to raise the BUSY signal within 3 micro-seconds on A2630s,
        otherwise data will be lost. Should be used only in an exclusive-
        access Open().

        A SLOWMODE mode can be specified (bit 4 of io_ParFlags) when very
        slow printers are used. If the printer acknowledges data at less
        than 5000 bytes per second, then this mode will actually save CPU
        time, although it consumes much more with high-speed printers.

        The PTermArray of the ioExtPar is initialized only if the EOFMODE
        bit (bit 1 of io_ParFlags) is set. The PTermArray can be further
        modified using the PDCMD_SETPARAMS command.

   INPUTS
        "parallel.device" - a pointer to literal string "parallel.device"
        unit - Must be zero for future compatibility
        ioExtPar - pointer to an IO Request block of structure IOExtPar to
                   be initialized by the Open() function. The io_ParFlags
                   field must be set as desired.
        flags - Must be zero for future compatibility

   RESULTS
        d0 -- same as io_Error
        io_Error -- if the Open succeded, then io_Error will be null.
                    If the Open failed, then io_Error will be non-zero.

   SEE ALSO
        exec/CloseDevice
```

```
parallel.device/PDCMD_QUERY                          parallel.device/PDCMD_QUERY

   NAME
        Query -- query parallel port/line status

   FUNCTION
        This command return the status of the parallel port lines and
        registers.

   IO REQUEST
        io_Message      must have mn_ReplyPort initialized
        io_Device       set by OpenDevice
        io_Unit         set by OpenDevice
        io_Command      PDCMD_QUERY (09)

   RESULTS
        io_Status        BIT  ACTIVE  FUNCTION

                          0    high   printer busy toggle (offline)
                          1    high   paper out
                          2    high   printer selected on the A1000
                                      printer selected & serial "Ring
                                      Indicator" on the A500/A2000
                                      Use care when making cables.
                          3     -     read=0,write=1
                        4-7           reserved

   BUGS
        In a earlier version of this AutoDoc, BUSY and PSEL were reversed.
        The function has always been correct.
```

```
parallel.device/PDCMD_SETPARAMS                 parallel.device/PDCMD_SETPARAMS

   NAME
        SetParams -- change parameters for the parallel device

   FUNCTION
        This command allows the caller to change the EOFMODE parameter for
        the parallel port device. It will disallow changes if any reads or
        writes are active or queued.

        The PARB_EOFMODE bit of io_ParFlags controlls whether the
        io_PTermArray is to be used as an additional termination criteria
        for reads and writes.  It may be set directly without a call to
        SetParams, setting it here performs the additional service of
        copying the PTermArray into the device default array which is used
        as the initial array for subsequent device opens. The Shared bit
        can be changed here, and overrides the current device access mode
        set at OpenDevice time.

   IO REQUEST
        io_Message      mn_ReplyPort initialized
        io_Device       preset by OpenDevice
        io_Unit         preset by OpenDevice
        io_Command      PDCMD_SETPARAMS (0A)
                        NOTE that the following fields of your IORequest
                        are filled by Open to reflect the parallel device's
                        current configuration.
        io_PExtFlags    must be set to zero, unless used
        io_ParFlags     see definition in parallel.i or parallel.h
                        NOTE that x00 yields exclusive access, PTermArray
                        inactive.
        io_PTermArray   ASCII descending-ordered 8-byte array of
                        termination characters. If less than 8 chars
                        used, fill out array w/lowest valid value.
                        Terminators are used only if EOFMODE bit of
                        io_Parflags is set. (e.g. x512F040303030303 )
                        This field is filled on OpenDevice only if the
                        EOFMODE bit is set.

   RESULTS
        io_Error -- if the SetParams succeded, then io_Error will be null.
                    if the SetParams failed, then io_Error will be non-zero.
```

printer.device Page 1

TABLE OF CONTENTS

printer.device/CMD_FLUSH
printer.device/CMD_INVALID
printer.device/CMD_RESET
printer.device/CMD_START
printer.device/CMD_STOP
printer.device/CMD_WRITE
printer.device/PRD_DUMPRPORT
printer.device/PRD_PRTCOMMAND
printer.device/PRD_QUERY
printer.device/PRD_RAWWRITE
printer.device/PWrite

printer.device Page 2

```
printer.device/CMD_FLUSH                                 printer.device/CMD_FLUSH

   NAME
        CMD_FLUSH -- abort all I/O requests (immediate)

   FUNCTION
        CMD_FLUSH aborts all stopped I/O at the unit.

   IO REQUEST
        io_Message      mn_ReplyPort set if quick I/O is not possible
        io_Device       preset by the call to OpenDevice
        io_Command      CMD_FLUSH
        io_Flags        IOB_QUICK set if quick I/O is possible
```

printer.device Page 3

```
printer.device/CMD_INVALID                          printer.device/CMD_INVALID

   NAME
       CMD_INVALID -- invalid command

   FUNCTION
       CMD_INVALID is always an invalid command, and sets the device
       error appropriately.

   IO REQUEST
       io_Message      mn_ReplyPort set if quick I/O is not possible
       io_Command      CMD_INVALID
       io_Flags        IOB_QUICK set if quick I/O is possible
```

printer.device Page 4

```
printer.device/CMD_RESET                                  printer.device/CMD_RESET

   NAME
       CMD_RESET -- reset the printer

   FUNCTION
       CMD_RESET resets the printer device without destroying handles
       to the open device.

   IO REQUEST
       io_Message      mn_ReplyPort set if quick I/O is not possible
       io_Device       preset by the call to OpenDevice
       io_Command      CMD_RESET
       io_Flags        IOB_QUICK set if quick I/O is possible
```

```
printer.device/CMD_START                           printer.device/CMD_START

   NAME
        CMD_START -- restart after stop (immediate)

   FUNCTION
        CMD_START restarts the unit after a stop command.

   IO REQUEST
        io_Message      mn_ReplyPort set if quick I/O is not possible
        io_Device       preset by the call to OpenDevice
        io_Command      CMD_START
        io_Flags        IOB_QUICK set if quick I/O is possible
```

```
printer.device/CMD_STOP                             printer.device/CMD_STOP

   NAME
        CMD_STOP -- pause current and queued I/O requests (immediate)

   FUNCTION
        CMD_STOP pauses all queued requests for the unit, and tries to
        pause the current I/O request.  The only commands that will
        be subsequently allowed to be performed are immediate I/O
        requests, which include those to start, flush, and finish the
        I/O after the stop command.

   IO REQUEST
        io_Message      mn_ReplyPort set if quick I/O is not possible
        io_Device       preset by the call to OpenDevice
        io_Command      CMD_STOP
        io_Flags        IOB_QUICK set if quick I/O is possible
```

```
printer.device/CMD_WRITE                               printer.device/CMD_WRITE

  NAME
     CMD_WRITE -- send output to the printer

  FUNCTION
     This function causes a buffer of characters to be written to the
     current printer port (usually parallel or serial).  The number of
     characters is specified in io_Length, unless -1 is used, in which
     case output is sent until a 0x00 is encountered.

     The Printer device, like the Console device, maps ANSI X3.64 style
     7-bit printer control codes to the control code set of the current
     printer.  The ANSI codes supported can be found below.

  NOTES
     Not all printers will support all functions.  In particular you may
     not assume that the MARGINS or TABS can be set.  Close to half the
     supported printers don't fully implement one or the other.  If you
     want the features of margins or tabs you will need to fake it
     internally by sending out spaces.

     Note that the printer device may have already sent out a "set
     margins" command to the printer.  If you are faking your own
     margins, be sure to cancel the old ones first.  (use the "aCAM"
     command)

     Defaults are set up so that if a normal AmigaDOS text file
     is sent to PRT:, it has the greatest chance of working.
     (AmigaDOS text files are defined as follows:)
             tabs            - every 8
             CR (0x0D)       - moves to start of current line
             LF (0x0A)       - moves to start of next line

  IO REQUEST
       io_Message      mn_ReplyPort set
       io_Device       preset by OpenDevice
       io_Unit         preset by OpenDevice
       io_Command      CMD_WRITE
       io_Length       number of characters to process, or if -1,
                       process until 0x00 encountered
       io_Data         pointer to block of data to process

  RESULTS
       io_Error : if CMD_WRITE succeeded, then io_Error will be zero.
              Otherwise io_Error will be non-zero.

  SEE ALSO
       printer.h, parallel.device, serial.device, Preferences


  ANSI X3.64 style COMMANDS

aRIS        ESCc          hard reset
aRIN        ESC#1         initialize to defaults
aIND        ESCD          true linefeed (lf)
aNEL        ESCE          return,lf
aRI         ESCM          reverse lf                  *

aSGR0       ESC[0m        normal character set
aSGR3       ESC[3m        italics on
aSGR23      ESC[23m       italics off
aSGR4       ESC[4m        underline on
aSGR24      ESC[24m       underline off
aSGR1       ESC[1m        boldface on
aSGR22      ESC[22m       boldface off
```

```
aSFC        SGR30-39        set foreground color
aSBC        SGR40-49        set background color

aSHORP0     ESC[0w          normal pitch
aSHORP2     ESC[2w          elite on
aSHORP1     ESC[1w          elite off
aSHORP4     ESC[4w          condensed on
aSHORP3     ESC[3w          condensed off
aSHORP6     ESC[6w          enlarged on
aSHORP5     ESC[5w          enlarged off

aDEN6       ESC[6"z         shadow print on
aDEN5       ESC[5"z         shadow print off
aDEN4       ESC[4"z         doublestrike on
aDEN3       ESC[3"z         doublestrike off
aDEN2       ESC[2"z         Near Letter Quality (NLQ) on
aDEN1       ESC[1"z         NLQ off

aSUS2       ESC[2v          superscript on
aSUS1       ESC[1v          superscript off
aSUS4       ESC[4v          subscript on
aSUS3       ESC[3v          subscript off
aSUS0       ESC[0v          normalize the line      *
aPLU        ESCL            partial line up         *
aPLD        ESCK            partial line down       *

aFNT0       ESC(B           US char set (default)   or Font 0
aFNT1       ESC(R           French char set         or Font 1
aFNT2       ESC(K           German char set         or Font 2
aFNT3       ESC(A           UK char set             or Font 3
aFNT4       ESC(E           Danish I char set       or Font 4
aFNT5       ESC(H           Sweden char set         or Font 5
aFNT6       ESC(Y           Italian char set        or Font 6
aFNT7       ESC(Z           Spanish char set        or Font 7
aFNT8       ESC(J           Japanese char set       or Font 8
aFNT9       ESC(6           Norweign char set       or Font 9
aFNT10      ESC(C           Danish II char set      or Font 10

aPROP2      ESC[2p          proportional on         *
aPROP1      ESC[1p          proportional off        *
aPROP0      ESC[0p          proportional clear      *
aTSS        ESC[n E         set proportional offset *
aJFY5       ESC[5 F         auto left justify       *
aJFY7       ESC[7 F         auto right justify      *
aJFY6       ESC[6 F         auto full justify       *
aJFY0       ESC[0 F         auto justify off        *
aJFY3       ESC[3 F         letter space (justify)  *
aJFY1       ESC[1 F         word fill(auto center)  *

aVERP0      ESC[0z          1/8" line spacing
aVERP1      ESC[1z          1/6" line spacing
aSLPP       ESC[nt          set form length n
aPERF       ESC[nq          set perforation skip to n lines (n>0)
aPERF0      ESC[0q          perforation skip off

aLMS        ESC#9           Left margin set         *
aRMS        ESC#0           Right margin set        *
aTMS        ESC#8           Top margin set          *
aBMS        ESC#2           Bottom margin set       *
aSTBM       ESC[Pn1;Pn2r    set T&B margins         *
aSLRM       ESC[Pn1;Pn2s    set L&R margin          *
aCAM        ESC#3           Clear margins

aHTS        ESCH            Set horiz tab           *
aVTS        ESCJ            Set vertical tabs       *
aTBC0       ESC[0g          Clr horiz tab           *
```

```
aTBC3       ESC[3g         Clear all h tab          *
aTBC1       ESC[1g         Clr vertical tabs        *
aTBC4       ESC[4g         Clr all v tabs           *
aTBCALL     ESC#4          Clr all h & v tabs       *
aTBSALL     ESC#5          Set default tabs (every 8)

aEXTEND     ESC[Pn"x       Extended commands
                           This is a mechanism for printer drivers to
                           support extra commands which can be called
                           by ANSI control sequences
aRAW        ESC[Pn"r       Next 'Pn' chars are raw (ie. they are not
                           parsed by the printer device, instead they
                           are sent directly to the printer.

(*) indicates that sending this command may cause unexpected results
    on a large number of printers.
```

```
printer.device/PRD_DUMPRPORT                         printer.device/PRD_DUMPRPORT

   NAME
        PRD_DUMPRPORT -- dump the specified RastPort to a graphics printer

   FUNCTION
        Print a rendition of the supplied RastPort, using the supplied
        ColorMap, position and scaling information, as specified in
        the printer preferences.

   IO REQUEST
        io_Message      mn_ReplyPort set if quick I/O is not possible.
        io_Command      PRD_DUMPRPORT.
        io_Flags        IOB_QUICK set if quick I/O is possible.
        io_RastPort     ptr to a RastPort.
        io_ColorMap     ptr to a ColorMap.
        io_Modes        the 'modes' flag from a ViewPort structure,
                        (the upper word is reserved and should be zero).

                        If you are running under version 36, or greater
                        of graphics.library, it is recommended that
                        you fill in "io_Modes" with the ULONG (32-bit)
                        value returned from calling:

                        ULONG ModeID = GetVPModeID(struct ViewPort *);

                        Doing so provides for upwards compatability with
                        the new display modes available under V36
                        (example: aspect ratio calculations for new
                        display modes).

        io_SrcX         x offset into the RastPort to start printing from.
        io_SrcY         y offset into the RastPort to start printing from.
        io_SrcWidth     width of the RastPort to print (from io_SrcX).
        io_SrcHeight    height of the RastPort to print (from io_SrcY).
        io_DestCols     width of the printout in printer pixels.
        io_DestRows     height of the printout in printer pixels.
        io_Special      flag bits
                        (some of which pertain to DestCols and DestRows).
                        -if SPECIAL_MIL is set, then the associated
                         parameter is specified in thousandths of
                         an inch on the printer.  ie. if DestCols = 8000,
                         DestRows = 10500 and SPECIAL_MILROWS and
                         SPECIAL_MILCOLS is set then the printout would be
                         8.000 x 10.500 inches.
                        -if SPECIAL_FULL is set, then the specific dimension
                         is set to the maximum possible as determined
                         by the printer limits or the configuration
                         limits; whichever is less.
                        -if SPECIAL_FRAC is set, the parameter is
                         taken to be a longword binary fraction
                         of the maximum for that dimension.
                        -if all bits for a dimension are clear,
                         (ie. SPECIAL MIL/FULL/FRAC and ASPECT are NOT set)
                         then the parameter is specified in printer pixels.
                        -if SPECIAL_CENTER is set then the image will be
                         put between the left and right edge of the paper.
                        -if SPECIAL_ASPECT is set, one of the dimensions
                         may be reduced/expanded to preserve the aspect
                         ratio of the print.
                        -SPECIAL_DENSITY(1-7) this allows for a maximum of 7
                         different print densities.  DENSITY1 is the lowest
                         density and the default.
                        -SPECIAL_NOFORMFEED - this allows for the mixing of
                         text and graphics or multiple graphic dumps on page
                         oriented printers (usually laser jet printers).
```

When this flag is set the page will not be ejected after a graphic dump. If you perform another graphic dump without this flag set OR close the printer after printing text after a graphic dump, the page will be ejected.

-if SPECIAL_TRUSTME is set then the printer specific driver is instructed to not issue a reset command before and after the dump. If this flag is NOT checked by the printer specific driver then setting this flag has no effect. Since we now recommend that printer driver writers no longer issue a reset command it is probably a safe idea to always set this flag when calling for a dump.

-if SPECIAL_NOPRINT is set then the following is done: Compute print size, set 'io_DestCols' and 'io_DestRows' in the calling program's 'IODRPReq' structure and exit, DON'T PRINT. This allows the calling program to see what the final print size would be in printer pixels. Note that it modifies the 'io_DestCols' and 'io_DestRows' fields of your 'IODRPReq' structure. It also sets the print density and updates the 'MaxXDots', 'MaxYDots', 'XDotsInch', and 'YDotsInch' fields of the 'PrinterExtendedData' structure.

There following rules for the interpretation of io_DestRows and io_DestCols that may produce unexpected results when they are not greater than zero and io_Special is zero. They have been retained for compatability. The user will not trigger these other rules with well formed usage of io_Special.

When io_Special is equal to 0, the following rules (from the V1.1 printer.device, and retained for compatibility reasons) take effect. Remember, these special rules are for io_DestRows and io_DestCols and only take effect if io_Special is 0).

a) DestCols>0 & DestRows>0 - use as absolute values. ie. DestCols=320 & DestRows=200 means that the picture will appear on the printer as 320x200 dots.
b) DestCols=0 & DestRows>0 - use the printers maximum number of columns and print DestRows lines. ie. if DestCols=0 and DestRows=200 than the picture will appear on the printer as wide as it can be and 200 dots high.
c) DestCols=0 & DestRows=0 - same as above except the driver determines the proper number of lines to print based on the aspect ratio of the printer. ie. This results in the largest picture possible that is not distorted or inverted. Note: As of this writing, this is the call made by such program as DeluxePaint, GraphicCraft, and AegisImages.
d) DestCols>0 &DestRows=0 - use the specified width and the driver determines the proper number of lines to print based on the aspect ratio of the printer. ie. if you desire a picture that is 500 pixels wide and aspect ratio correct, use DestCols=500 and DestRows=0.
e) DestCols<0 or DestRows>0 - the final picture is either a reduction or expansion based on the fraction |DestCols| / DestRows in the proper aspect ratio.
Some examples:
1) if DestCols=-2 & DestRows=1 then the printed picture will be 2x the AMIGA picture and in the proper aspect ratio. (2x is derived from |-2| / 1 which gives 2.0)
2) if DestCols=-1 & DestRows=2 then the printed picture will will be 1/2x the AMIGA picture in the proper aspect ratio. (1/2x is derived from |-1| / 2 which gives 0.5)

NOTES

The printer selected in preferences must have graphics capability to use this command. The error 'PDERR_NOTGRAPHICS' is returned if the printer can not print graphics.

Color printers may not be able to print black and white or greyscale pictures -- specifically, the Okimate 20 cannot print these with a color ribbon: you must use a black ribbon instead. If the printer has an input buffer option, use it. If the printer can be uni or bi directional, select uni-directional; this produces a much cleaner picture. Most printer drivers will attempt to set unidirectional printing if it is possible under software control.

Please note that the width and height of the printable area on the printer is in terms of pixels and bounded by the following:
a) WIDTH = (RIGHT_MARGIN - LEFT_MARGIN + 1) / CHARACTERS_PER_INCH
b) HEIGHT = LENGTH / LINES_PER_INCH
Margins are set by preferences.

For BGR printer support, the YMC values in the printer specific render.c functions equate to BGR respectively, ie. yellow is blue, magenta is green, and cyan is red.

Data Structures

The printer specific and non-specific data structures can be read ONCE you have opened the printer device. Here is a code fragment to illustrate how to do just that.

```
        #include <exec/types.h>
        #include <devices/printer.h>
        #include <devices/prtbase.h>
        #include <devices/prtgfx.h>

        struct IODRPReq PReq;
        struct PrinterData *PD;
        struct PrinterExtendedData *PED;

   open the printer device / if it opened...
   if (OpenDevice("printer.device", 0, &PReq, 0) == NULL) {
       get pointer to printer data
       PD = (struct PrinterData *)PReq.io_Device;
       get pointer to printer extended data
       PED = &PD->pd_SegmentData->ps_PED;
       let's see what's there
       printf("PrinterName = '%s', Version=%u, Revision=%u\n",
           PED->ped_PrinterName, PD->pd_SegmentData->ps_Version,
           PD->pd_SegmentData->ps_Revision,);
       printf("PrinterClass=%u, ColorClass=%u\n",
           PED->ped_PrinterClass, PED->ped_ColorClass);
       printf("MaxColumns=%u, NumCharSets=%u, NumRows=%u\n",
           PED->ped_MaxColumns, PED->ped_NumCharSets, PED->ped_NumRows);
       printf("MaxXDots=%lu, MaxYDots=%lu, XDotsInch=%u, YDotsInch=%u\n",
           PED->ped_MaxXDots, PED->ped_MaxYDots,
           PED->ped_XDotsInch, PED->ped_YDotsInch);
       CloseDevice(&PReq);
   }
```

Preferences

If you want the user to be able to access the printer preferences items without having to run preferences (like DPAINT II's printer requester), here is what you do. You can look at the printer's copy of preferences

```
by referring to 'PD->pd_Preferences' (the printer device MUST already be
opened at this point).  After you have this you could put up a requester
and allow the user to change whatever parameters they wanted.
BEAR IN MIND THAT YOU ARE RESPONSIBLE FOR RANGE CHECKING THESE SELECTIONS!
Listed below are the printer preferences items and their valid values.

PrintPitch          - PICA, ELITE, FINE.
PrintQuality        - DRAFT, LETTER.
PrintSpacing        - SIX_LPI, EIGHT_LPI.
PrintLeftMargin     - 1 to PrintRightMargin.
PrintRightMargin    - PrintLeftMargin to 999.
PaperLength         - 1 to 999.
PrintImage          - IMAGE_POSITIVE, IMAGE_NEGATIVE.
PrintAspect         - ASPECT_HORIZ, ASPECT_VERT.
PrintShade          - SHADE_BW, SHADE_GREYSCALE, SHADE_COLOR.
PrintThreshold      - 1 to 15.
PrintFlags          - CORRECT_RED, CORRECT_GREEN, CORRECT_BLUE, CENTER_IMAGE,
                      IGNORE_DIMENSIONS, BOUNDED_DIMENSIONS,
                      ABSOLUTE_DIMENSIONS, PIXEL_DIMENSIONS,
                      MULTIPLY_DIMENSIONS, INTEGER_SCALING,
                      ORDERED_DITHERING, HALFTONE_DITHERING.
                      FLOYD_DITHERING, ANTI_ALIAS, GREY_SCALE2
PrintMaxWidth       - 0 to 65535.
PrintMaxHeight      - 0 to 65535.
PrintDensity        - 1 to 7.
PrintXOffset        - 0 to 255.

Asynchronous I/O
----------------

    The recommended way to do asynchronous i/o is...

a) To send requests for i/o.

    struct IORequest *ioreq;
    struct MsgPort *port;
    UBYTE signal;

    port = ioreq->io_Message.mn_ReplyPort;
    signal = port->mp_SigBit;

    SendIO(ioreq);  send request
    Wait(signal);  wait for completion (go to sleep)
    while ((Msg = GetMsg(port)) != NULL) {  get ALL messages
    }

b) To abort a previous request for i/o.

    struct IORequest *ioreq;

    AbortIO(ioreq);  abort request
    WaitIO(ioreq);  wait for reply

    at this point you can re-use 'ioreq'.

    Note that in the above examples 'ioreq' could be any one of...
    a) struct IOStdReq    a standard i/o request
    b) struct IODRPReq    a dumprport i/o request
    c) struct IOPrtCmdReq a printer command i/o request

    It is recommend that you do asynchronous i/o in your programs
    and give the user a way of aborting all requests.


               V1.3 Printer Driver Notes
               -------------------------
```

```
    In general densities which use more than one pass should only be
used for B&W shade dumps.  They can be used for Grey-Scale or Color Shade
dumps BUT the output may tend to look muddy or dark.  Also multiple pass
Color dumps tend to dirty or smear the ribbon (ie. yellow will get
contaminated with the other colors on the ribbon; you have been warned).


Alphacom_AlphaPro_101
---------------------
1. Daisywheel printer (text only).

Brother_HR-15XL
---------------
1. Daisywheel printer (text only).

CalComp_ColorMaster
-------------------
1. Thermal transfer b&w/color printer (text and graphics).
2. Use Black ribbon for non-color dumps; Color ribbon for color dumps.
3. Linefeeds # of vertical dots printed.
4. Densities supported are 203x200(1) dpi.
5. This is a dual printer driver.  Select a PaperSize of 'Narrow Tractor'
   for use with the ColorMaster; 'Wide Tractor' for use with the
   ColorView-5912 (which uses 11 x 17 inch paper).

CalComp_ColorMaster2
--------------------
1. Thermal transfer b&w/color printer (text and graphics).
2. Use Black ribbon for non-color dumps; Color ribbon for color dumps.
3. Linefeeds # of vertical dots printed.
4. Densities supported are 203x200(1) dpi.
5. This is a dual printer driver.  Select a PaperSize of 'Narrow Tractor'
   for use with the ColorMaster; 'Wide Tractor' for use with the
   ColorView-5912 (which uses 11 x 17 inch paper).
6. This driver is the same as the Calcomp_ColorMaster driver EXCEPT it is
   approximately 2 times faster (during color dumps) and requires LOTS of
   memory (up to 1,272,003 bytes for a full 8 x 10 inch (1600 x 2000 dot)
   color dump).  Typically full-size (color) dumps are 1600 x 1149 dots and
   require 730,767 bytes.  Memory requirements for the ColorView-5912
   are up to 2,572,803 bytes for a full 10 x 16 inch (2048 x 3200 dot)
   color dump.  Typically full-size (color) dumps are 2048 x 2155 dots and
   require 1,732,623 bytes.  The memory requirements are 1/3 when doing a
   non-color printout (on both the ColorMaster and ColorView).

Canon_PJ-1080A
--------------
1. Ink jet b&w/color printer (text and graphics).
2. Linefeeds # of vertical dots printed.
3. Densities supported are 83x84(1) dpi.

CBM_MPS1000
-----------
1. Dot matrix b&w printer (text and graphics).
2. Linefeeds # of vertical dots printed (-1/3 dot if PaperType = Single). *2
3. Density    XDPI   YDPI   XYDPI   Comments
       1      120     72     8640
       2      120    144    17280   two pass
       3      240     72    17280                 *1
       4      120    216    25920   three pass
       5      240    144    34560   two pass      *1
       6      240    216    51840   three pass    *1
       7      same as 6

Diablo_630
----------
```

```
1. Daisywheel printer (text only).

Diablo_Advantage_D25
--------------------
1. Daisywheel printer (text only).

Diablo_C-150
------------
1. Ink jet b&w/color printer (text and graphics).
2. Always linefeeds 4 dots (limitation of printer).
3. A PaperSize of 'Wide Tractor' selects a maximum print width of
   8.5 inches (for wide roll paper).
5. Densities supported are 120x120(1) dpi.

EpsonQ (24-pin Epson compatible)
------
1. Dot matrix b&w/color printer (text and graphics).
2. Drives all EpsonQ (LQ1500, LQ2500, etc.) compatible printers.
3. Linefeeds # of vertical dots printed.
4. Density      XDPI     YDPI     XYDPI    Comments
        1        90      180      16200
        2       120      180      21600
        3       180      180      32400
        4       360      180      64800    *1
        5,6,7   same as 4
5. A PaperSize of 'Wide Tractor' selects a maximum print width of
   13.6 inches (for wide carriage printers).
6. A PaperType of 'Single' uses only 16 of the 24 pins, whereas a PaperType
   of 'Fanfold' uses all 24 pins.  The 'Single' option is useful for those
   printers which have a weak power supply and cannot drive all 24 pins
   continuously.  If during a single pass of the print head you notice that
   the top two thirds of the graphics are darker than the bottom one third
   then you will probably need to drop down to 16 pins.

EpsonX[CBM_MPS-1250] (8/9-pin Epson compatible)
--------------------
1. Dot matrix b&w/color printer (text and graphics).
2. Drives all EpsonX (EX/FX/JX/LX/MX/RX, etc.) compatible printers.
3. Linefeeds # of vertical dots printed (-1/3 dot if PaperType = Single). *2
4. Density      XDPI     YDPI     XYDPI    Comments
        1       120       72       8640
        2       120      144      17280    two pass
        3       240       72      17280                    *1
        4       120      216      25920    three pass
        5       240      144      34560    two pass        *1
        6       240      216      51840    three pass      *1
        7       same as 6

5. A PaperSize of 'Wide Tractor' selects a maximum print width of
   13.6 inches (for wide carriage printers).
6. Use this driver if you own a CBM_MPS-1250 (as it is EpsonX compatible).

EpsonXOld (8/9-pin Epson compatible)
---------
1. Dot matrix b&w printer (text and graphics).
2. Drives all very old EpsonX (EX/FX/JX/LX/MX/RX, etc.) compatible printers.
3. Linefeeds # of vertical dots printed.
4. Density      XDPI     YDPI     XYDPI    Comments
        1        60      72        4320
        2       120      72        8640    (double speed)               *1
        3       120      72        8640
        4       240      72       17280                                 *1
        5       120      72        8640    (for use on old Star printers)
        6       240      72       17280    (for use on old Star printers)  *1
        7       240      72       17280    (same as density 4)          *1
5. A PaperSize of 'Wide Tractor' selects a maximum print width of
```

```
   13.6 inches (for wide carriage printers).
6. Use this driver if the EpsonX driver does not work properly in graphics
   or text mode on your EpsonX compatible printer.

generic
-------
1. Text only printer.

Howtek_Pixelmaster
------------------
1. Plastic ink jet b&w/color printer (text and graphics).
2. Linefeeds # of vertical dots printed.
3. Density      XDPI     YDPI     XYDPI    Comments
        1        80       80       6400
        2       120      120      14400
        3       160      160      25600
        4       240      240      57600
        5,6,7   same as 4
4. Maximum print area is 8.0 x 10.0 inches.

HP_DeskJet
----------
1. Ink jet non-color printer (text and graphics).
2. Linefeeds # of vertical dots printed.
3. Density      XDPI     YDPI     XYDPI    Comments
        1        75       75       5625
        2       100      100      10000
        3       150      150      22500
        4       300      300      90000
        5,6,7   same as 4
4. Maximum print area is 8.0 x 10.0 inches.

HP_LaserJet (LaserJet+/LaserJetII compatible)
-----------
1. Laser engine non-color printer (text and graphics).
2. Linefeeds # of vertical dots printed.
3. Density      XDPI     YDPI     XYDPI    Comments
        1        75       75       5625
        2       100      100      10000
        3       150      150      22500
        4       300      300      90000
        5,6,7   same as 4
4. Maximum print area is 8.0 x 10.0 inches.

HP_PaintJet
-----------
1. Ink jet b&w/color printer (text and graphics).
2. Linefeeds # of vertical dots printed.
3. Densities supported are 180x180(1) dpi.

HP_ThinkJet
-----------
1. Ink jet non-color printer (text and graphics).
2. Linefeeds # of vertical dots printed.
3. Density      XDPI     YDPI     XYDPI    Comments
        1        96       96       9216
        2       192       96      18432
        3,4,5,6,7         same as 4

Imagewriter II (Imagewriter compatible)
--------------
1. Dot matrix b&w/color printer (text and graphics).
2. Linefeeds # of vertical dots printed.
3. Density      XDPI     YDPI     XYDPI    Comments
        1        80       72       5760
        2       120       72       8640
        3       144     72      10368
        4       160     72      11520
        5       120     144     17280   two pass
        6       144     144     20736   two pass
        7       160     144     23040   two pass

Nec_Pinwriter (24-wire Pinwriter compatible (P5/P6/P7/P9/P2200))
-------------
1. Dot matrix b&w/color printer (text and graphics).
2. Drives all Nec 24-wire Pinwriter compatible printers.
3. Linefeeds # of vertical dots printed.
4. Density      XDPI    YDPI    XYDPI   Comments
        1        90     180      16200
        2       120     180      21600
        3       180     180      32400
        4       120     360      43200  two pass
        5       180     360      64800  two pass
        6       360     180      64800
        7       360     360     129600  two pass
5. A PaperSize of 'Wide Tractor' selects a maximum print width of
   13.6 inches (for wide carriage printers).

Okidata_92
----------
1. Dot matrix non-color printer (text and graphics).
2. Always linefeeds 7/72 inch (limitation of printer in graphics mode).
3. Densities supported are 72x72 dpi.

Okidata_293I
------------
1. Dot matrix b&w/color printer (text and graphics).
2. Drives 292 or 293 using the IBM interface module.
3. Linefeeds # of vertical dots printed (-1/2 dot if PaperType = Single) *3
4. Density      XDPI    YDPI    XYDPI   Comments
        1       120     144     17280
        2       240     144     34560
        3       120     288     34560   two pass
        4       240     288     69120   two pass
        5,6,7   same as 4
5. A PaperSize of 'Wide Tractor' selects a maximum print width of
   13.6 inches (for wide carriage printers).

Okimate-20
----------
1. Thermal transfer b&w/color printer (text and graphics).
2. Use Black ribbon for non-color dumps; Color ribbon for color dumps.
3. Linefeeds an even # of dots printed. (ie. if 3 printed, 4 advanced).
4. Densities supported are 120x144(1) dpi.

Quadram_QuadJet
---------------
1. Ink jet b&w/color printer (text and graphics).
2. Linefeeds # of vertical dots printed.
3. Densities supported are 83x84(1) dpi.

Qume_LetterPro_20
-----------------
1. Daisywheel printer (text only).

Seiko_5300
----------
1. Thermal transfer b&w/color printer (graphics only).
2. Use Black ribbon for non-color dumps; Color ribbon for color dumps.
3. Density      XDPI    YDPI    XYDPI   Comments
        1       152     152     23104   drives CH-5301 printer
        2       203     203     41209   drives CH-5312 printer
```

```
        3       240     240     57600   drives CH-5303 printer
        4, 5,6,7        same as 3
        You must select the proper density to drive the specific printer
        that you have.
4. This driver is not on the V1.3 Workbench or Extras disk.  It is
   available on BIX and directly from Seiko.

Seiko_5300a
-----------
1. Thermal transfer b&w/color printer (graphics only).
2. Use Black ribbon for non-color dumps; Color ribbon for color dumps.
3. Density      XDPI    YDPI    XYDPI   Comments
        1       152     152     23104   drives CH-5301 printer
        2       203     203     41209   drives CH-5312 printer
        3       240     240     57600   drives CH-5303 printer
        4, 5,6,7        same as 3
        You must select the proper density to drive the specific printer
        that you have.
4. This driver is the same as the Seiko_5300 driver EXCEPT it is
   approximately 2 times faster (during color dumps) and requires LOTS of
   memory (up to 1,564,569 bytes for a full 8 x 10 inch (1927 x 2173 dot)
   color dump).  Typically full-size (color) dumps are 1927 x 1248 dots
   and require 898,569 bytes.  The memory requirements are 1/3 when doing
   a non-color printout.
5. This driver is not on the V1.3 Workbench or Extras disk.  It is
   available on BIX and directly from Seiko.

Tektronix_4693D
---------------
1. Thermal transfer b&w/color printer (graphics only).
2. Densities supported are 300x300(1) dpi
3. Due to the way the printer images a picture none of the printer
   preferences options affect the printout with the following exceptions:
   a)Aspect - Horizontal, Vertical
   b)Shade - B&W, Grey_Scale, Color
   ...as a result of this only full size pictures can be printed.
4. Keypad menu option 3b COLOR ADJUSTMENT may be set from the keypad.
   For normal prints this option should be set to "do not adjust".
5. Keypad menu option 3d VIDEO COLOR CORRECTION may be set from the keypad.
   For normal prints this option should be set to "do not adjust".
6. Keypad menu option 5 BACKGROUND COLOR EXCHANGE may be set from the
   keypad.  For normal prints this option should be set to "print colors
   as received".
7. Once a picture has been printed additional copies may be printed
   without resending by using the printers keypad.
8. This driver is not on the V1.3 Workbench or Extras disk.  It is
   available on BIX and directly from Tektronix.

Tektronix_4696
--------------
1. Ink jet b&w/color printer (text and graphics).
2. Always linefeeds 4 dots (limitation of printer).
3. Densities supported are 121x120(1), 242x120(black)(2) and
   242x120(color)(3).
   Selecting a density of 2 or higher really doesn't give you true 242 dpi
   resolution since the printer only has 121 x dots per inch.
   Instead this mode tells the printer to go into it's double pass mode.
   Here, it outputs a line of dots at 121 dpi; and outputs the line again
   (shifted to the right by 1/242 of an inch).  This produces much more
   vibrate colors and gives the illusion of more resolution.  One drawback
   is that large areas of solid colors (red, green, and blue specifically)
   tend to over-saturate the paper with ink.  Density1 outputs all colors
   in one pass.  Density 2 does a double pass on black.  Density 3 does a
   double pass on all colors.  Density 1 to 3 correspond to the printer's
   graphics printing modes 1 to 3 (respectively).
4. This driver is not on the V1.3 Workbench or Extras disk.  It is
```

```
   available on BIX and directly from Tektronix.
5. A PaperSize of 'Wide Tractor' selects a maximum print width of
   9.0 inches (for wide roll paper).

Toshiba_P351C (24-pin Toshiba compatible)
-------------
1. Dot matrix b&w/color printer (text and graphics).
2. Drives all Toshiba_P351C compatible printers.
3. Linefeeds # of vertical dots printed.
4. Density      XDPI    YDPI    XYDPI   Comments
        1       180     180     32400
        2       360     180     64800
        3,4,5,6,7       same as 2
5. A PaperSize of 'Wide Tractor' selects a maximum print width of
   13.5 inches (for wide carriage printers).

Toshiba_P351SX (24-pin Toshiba compatible)
--------------
1. Dot matrix b&w/color printer (text and graphics).
2. Drives all Toshiba_P351SX (321SL, 321SLC, 341SL) compatible printers.
3. Linefeeds # of vertical dots printed.
4. Density      XDPI    YDPI     XYDPI   Comments
        1       180     180      32400
        2       360     180      64800
        3       180     360      64800   two pass
        4       360     360     129600   two pass
        5,6,7   same as 4
5. A PaperSize of 'Wide Tractor' selects a maximum print width of
   13.5 inches (for wide carriage printers).

Xerox_4020
----------
1. Ink jet b&w/color printer (text and graphics).
2. Always linefeeds 4 dots (limitation of printer).
3. This driver is IDENTICAL to the Diablo_C-150 driver EXCEPT it outputs
   all black dots TWICE.  This is a special feature of this printer and
   produces much more solid, darker black shades.  Please note that some
   printing time overhead results from this feature; if you don't want it
   use the Diablo_C-150 driver.
4. Densities supported are 121x120(1) and 242x240(2) dpi.
   Selecting a density of 2 or higher really doesn't give you true 240 dpi
   resolution since the Xerox_4020 only has 121 x dots per inch.
   Instead this mode tells the printer to go into it's pseudo 240 dpi mode.
   Here, it outputs a line of dots at 121 dpi; moves the paper up 1/240 of
   an inch and outputs the line again (shifted to the right by 1/240 of an
   inch).  This produces much more vibrate colors and gives the illusion
   of more resolution.  One drawback is that large areas of solid colors
   (red, green, and blue specifically) tend to over-saturate the paper with
   ink.
5. A PaperSize of 'Wide Tractor' selects a maximum print width of
   9.0 inches (for wide roll paper).

Notes
-----

*0 - on most printers friction fed paper tends to produce better looking
     (ie. less horizontal banding) graphic dumps than tractor fed paper.

*1 - in this mode the printer cannot print two consecutive dots in a row.
     It is recommended that you only use this density for B&W Shade dumps.

*2 - only when 72 YDPI is selected.  This option is useful if you notice
     tiny white horizontal strips in your printout.

*3 - only when 144 YDPI is selected.  This option is useful if you notice
     tiny white horizontal strips in your printout.
```

```
printer.device/PRD_PRTCOMMAND                    printer.device/PRD_PRTCOMMAND

   NAME
       PRD_PRTCOMMAND -- send a command to the printer

   FUNCTION
      This function sends a command to either the parallel or serial
      device.  The printer device maps this command to the control
      code set of the current printer.  The commands supported can
      be found with the printer.device/Write command.  All printers
      may not support all functions.

   IO REQUEST IOPrtCmdReq
       io_Message      mn_ReplyPort set
       io_Device       preset by OpenDevice
       io_Unit         preset by OpenDevice
       io_Command      PRD_PRTCOMMAND
       io_PrtCommand   the actual command number
       io_Parm0        parameter for the command
       io_Parm1        parameter for the command
       io_Parm2        parameter for the command
       io_Parm3        parameter for the command

   RESULTS
       Errors: if the PRD_PRTCOMMAND succeeded, then io_Error will be zero.
       Otherwise io_Error will be non-zero.  An error of -1 indicates that
       the command is not supported by the current printer driver.  This
       could be used to check if the connected printer supports a particular
       command (italics for example).

   SEE ALSO
       printer.device/Write printer.h, parallel.device, Preferences
printer.device/PRD_QUERY                              printer.device/PRD_QUERY

   NAME
        PRD_QUERY -- query printer port/line status

   FUNCTION
        This command returns the status of the printer port's lines and
        registers.  Since the printer port uses either the serial or
        parallel port for i/o, the actual status returned is either the
        serial or parallel port's status.

   IO REQUEST
        io_Message      mn_ReplyPort set if quick I/O is not possible
        io_Device       preset by the call to OpenDevice
        io_Command      PRD_QUERY
        io_Data         ptr to 2 UBYTES where result will be stored.

   RESULTS
     io_Data           BIT  ACTIVE  FUNCTION (SERIAL DEVICE)

             LSB        0    low    reserved
                        1    low    reserved
                        2    low    reserved
                        3    low    Data Set Ready
                        4    low    Clear To Send
                        5    low    Carrier Detect
                        6    low    Ready To Send
                        7    low    Data Terminal Ready
             MSB        8    high   read buffer overflow
                        9    high   break sent (most recent output)
                       10    high   break received (as latest input)
                       11    high   transmit x-OFFed
                       12    high   receive x-OFFed
                    13-15           reserved


     io_Data           BIT  ACTIVE  FUNCTION (PARALLEL DEVICE)

                        0     hi     printer busy (offline)
                        1     hi     paper out
                        2     hi     printer selected
                                      (WARNING: the bit 2 line is also connected
                                       to the serial port's ring indicator pin
                                       on the A500 and A2000)
                      3-7            reserved

     io_Actual         1-parallel, 2-serial
```

```
printer.device/PRD_RAWWRITE                        printer.device/PRD_RAWWRITE

   NAME
        PRD_RAWWRITE -- transparent write command

   FUNCTION
        This is a non standard write command that performs no
        processing on the data passed to it.

   IO REQUEST
        io_Message      mn_ReplyPort set if quick I/O is not possible
        io_Command      PRD_RAWWRITE
        io_Flags        IOB_QUICK set if quick I/O is possible
        io_Length       the number of bytes in io_Data
        io_Data         the raw bytes to write to the printer
```

```
printer.device/PWrite                                    printer.device/PWrite

   NAME
        PWrite -- internal write to printer port

   SYNOPSIS
        error = (*PrinterData->pd_PWrite)(buffer, length);
        D0                                    A0      D0

   FUNCTION
        PWrite writes 'length' bytes directly to the printer.  This
        function is generally called by printer drivers to send
        their buffer(s) to the printer.

        This function is accessed by referencing off the PrinterData (PD)
        structure.  Below is a code fragment to show how to do get access
        to a pointer to the PrinterData (PD) structure.

        #include <exec/types.h>
        #include <devices/printer.h>
        #include <devices/prtbase.h>

        struct IODRPReq PReq;
        struct PrinterData *PD;
        struct PrinterExtendedData *PED;

        /* open the printer device (any version); if it opened... */
        if (OpenDevice("printer.device", 0, &PReq, 0) == NULL) {

            /* get pointer to printer data structure */
            PD = (struct PrinterData *)PReq.io_Device;

            /* write something directly to the printer */
            (*PD->pd_PWrite)("Hello world\n", 12);

            CloseDevice(&PReq); /* close the printer device */
        }
```

TABLE OF CONTENTS

serial.device/AbortIO
serial.device/BeginIO
serial.device/CloseDevice
serial.device/CMD_CLEAR
serial.device/CMD_FLUSH
serial.device/CMD_READ
serial.device/CMD_RESET
serial.device/CMD_START
serial.device/CMD_STOP
serial.device/CMD_WRITE
serial.device/OpenDevice
serial.device/SDCMD_BREAK
serial.device/SDCMD_QUERY
serial.device/SDCMD_SETPARAMS

```
serial.device/AbortIO                                  serial.device/AbortIO

   NAME
       AbortIO(ioRequest) -- abort an I/O request
               A1

   FUNCTION
      This is an exec.library call.

      This function attempts to aborts a specified read or write request.
      If the request is active, it is stopped immediately. If the request is
      queued, it is painlessly removed.  The request will be returned
      in the same way any completed request it.

      After AbortIO(), you must generally do a WaitIO().

   INPUTS
      iORequest  -- pointer to the IORequest Block that is to be aborted.

   RESULTS
       io_Error -- if the Abort succeded, then io_Error will be #IOERR_ABORTED

           (-2) and the request will be flagged as aborted (bit 5 of
           io_Flags is set).  If the Abort failed, then the Error will be zero.

   BUGS
       Previous to version 34, the serial.device would often hang when
       aborting CTS/RTS handshake requests.  This was the cause of the
       incorrect assumption that AbortIO() does not need to be followed
       by a wait for a reply (or a WaitIO()).
```

serial.device Page 3

```
serial.device/BeginIO                                    serial.device/BeginIO

   NAME
       BeginIO(ioRequest),deviceNode -- start up an I/O process
               A1         A6
   FUNCTION
       This is a direct function call to the device.  It is intended for
       more advanced programmers.  See exec's DoIO() and SendIO() for
       the normal method of calling devices.

       This function initiates a I/O request made to the serial
       device. Other than read or write, the functions are performed
       synchronously, and do not depend on any interrupt handling
       logic (or it's associated discontinuities), and hence should
       be performed as IO_QUICK.
       With some exceptions, reads and writes are merely initiated by
       BeginIO, and thusly return to the caller as begun, not completed.
       Completion is signalled via the standard ReplyMsg routine.
       Multiple requests are handled via FIFO queueing.
       One exception to this non-QUICK handling of reads and writes
       is for READS when:
         - IO_QUICK bit is set
         - There are no pending read requests
         - There is already enough data in the input buffer to satisfy
           this I/O Request immediately.
       In this case, the IO_QUICK flag is not cleared, and the request
       is completed by the time it returns to the caller. There is no
       ReplyMsg or signal bit activity in this case.

   INPUTS
       ioRequest  -- pointer to an I/O Request Block of size
           io_ExtSerSize (see serial.i for size/definition),
           containing a valid command in io_Command to process,
           as well as the command's other required parameters.
       deviceNode -- pointer to the "serial.device", as found in
           the IO_DEVICE of the ioRequest.

   RESULTS
       io_Error -- if the BeginIO succeded, then Error will be null.
           If the BeginIO failed, then the Error will be non-zero.
           I/O errors won't be reported until the io completes.

   SEE ALSO
        devices/serial.h
```

serial.device Page 4

```
serial.device/CloseDevice                          serial.device/CloseDevice

   NAME
       CloseDevice -- close the serial port

   SYNOPSIS
       CloseDevice(deviceNode)
                   A1
   FUNCTION
       This is an exec call that terminates communication with the
       serial device.  Upon closing, the device's input buffer is freed.

       Note that all IORequests MUST be complete before closing.
        If any are pending, your program must AbortIO() then WaitIO()
       to complete them.

   INPUTS
       deviceNode - pointer to the device node, set by Open

   SEE ALSO
       serial.device/OpenDevice
```

serial.device Page 5

```
serial.device/CMD_CLEAR                                  serial.device/CMD_CLEAR

   NAME
       Clear -- clear the serial port buffers

   FUNCTION
       This command resets the serial port's read buffer pointers.

   IO REQUEST
       io_Message      mn_ReplyPort initialized
       io_Device       set by OpenDevice
       io_Unit         set by OpenDevice
       io_Command      CMD_CLEAR

   RESULTS
       Error -- If the Clear succeded, then io_Error will be null.
                If the Clear failed, then the io_Error will be non-zero.
```

serial.device Page 6

```
serial.device/CMD_FLUSH                                  serial.device/CMD_FLUSH

   NAME
       Flush -- clear all queued I/O requests for the serial port

   FUNCTION
       This command purges the read and write request queues for the
       serial device. Flush will not affect active requests.

   IO REQUEST
       io_Message      mn_ReplyPort initialized
       io_Device       set by OpenDevice
       io_Unit         set by OpenDevice
       io_Command      CMD_FLUSH

   RESULTS
       Error -- if the Flush succeded, then io_Error will be null.
                If the Flush failed, then the io_Error will be non-zero.
```

serial.device

Page 7

```
serial.device/CMD_READ                                    serial.device/CMD_READ

   NAME
       Read -- read input from serial port

   FUNCTION
        This command causes a stream of characters to be read in from
        the serial port buffer.  The number of characters is specified
        in io_Length.

        The Query function can be used to check how many characters
        are currently waiting in the serial port buffer.  If more characters
        are requested than are currently available, the ioRequest
        will be queued until it can be satisfied.


        The best way to handle reads is to first Query to get the number
        of characters currently in the buffer.  Then post a read request
        for that number of characters (or the maximum size of your buffer).

        If zero characters are in the buffer, post a request
        for 1 character.  When at least one is ready, the device will return
        it.  Now start over with another Query.

        Before the program exits, it must be sure to AbortIO() then WaitIO()
        any outstanding ioRequests.

   IO REQUEST
       io_Message      A mn_ReplyPort is required
       io_Device       set by OpenDevice
       io_Unit         set by OpenDevice
       io_Command      CMD_READ
       io_Flags        If the IOB_QUICK bit is set, read will try
                        to complete the IO quickly
       io_Length       number of characters to receive.
       io_Data         pointer to buffer

   RESULTS
       Error -- if the Read succeded, then io_Error will be null.
           If the Read failed, then io_Error will be non-zero.
             io_Error will indicate problems such as parity mismatch,
             break, and buffer overrun.

   SEE ALSO
        serial.device/SDCMD_QUERY
        serial.device/SDCMD_SETPARAMS

   BUGS
        Having multiple outstanding read IORequests at any one time will
        probably fail.

        Old documentation mentioned a mode where io_Length was set to -1.
        If you want a NULL terminated read, use the io_TermArray instead.
```

serial.device

Page 8

```
serial.device/CMD_RESET                                  serial.device/CMD_RESET

   NAME
       Reset -- reinitializes the serial port

   FUNCTION
       This command resets the serial port to its freshly initialized
       condition. It aborts all I/O requests both queued and current,
       relinquishes the current buffer, obtains a new default sized
       buffer, and sets the port's flags and parameters to their
       boot-up time default values. The functions places the reset
       parameter values in the ioRequest block.

   IO REQUEST
       io_Message      mn_ReplyPort initialized
       io_Device       set by OpenDevice
       io_Unit         set by OpenDevice
       io_Command      CMD_RESET

   RESULTS
       Error -- if the Reset succeded, then Error will be null.
           If the Reset failed, then the Error will be non-zero.
```

```
serial.device/CMD_START                                 serial.device/CMD_START

   NAME
       Start -- restart paused I/O over the serial port

   FUNCTION
       This function restarts all current I/O on the serial port by
       sending an xON to the "other side", and submitting a "logical
       xON" to "our side", if/when appropriate to current activity.

   IO REQUEST
       io_Message      mn_ReplyPort initialized
       io_Device       set by OpenDevice
       io_Unit         set by OpenDevice
       io_Command      CMD_START

   RESULTS

   SEE ALSO
       serial.device/CMD_STOP
```

```
serial.device/CMD_STOP                                   serial.device/CMD_STOP

   NAME
       Stop -- pause all current I/O over the serial port

   FUNCTION
       This command halts all current I/O on the serial port by
       sending an xOFF to the "other side", and submitting a "logical
       xOFF" to "our side", if/when appropriate to current activity.

   IO REQUEST
       io_Message      mn_ReplyPort initialized
       io_Device       set by OpenDevice
       io_Unit         set by OpenDevice
       io_Command      CMD_STOP

   RESULTS

   SEE ALSO
       serial.device/CMD_START
```

serial.device Page 11

```
serial.device/CMD_WRITE                                 serial.device/CMD_WRITE

   NAME
       Write -- send output to serial port

   FUNCTION
       This command causes a stream of characters to be written out
       the serial port. The number of characters is specified in
       io_Length, unless -1 is used, in which case output is sent until
       a null(0x00) is encountered.

   IO REQUEST
       io_Message      must have mn_ReplyPort initialized
       io_Device       set by OpenDevice
       io_Unit         set by OpenDevice
       io_Command      CMD_WRITE
       io_Flags        Set IOF_QUICK to try quick I/O
       io_Length       number of characters to transmit, or if set
                       to -1 transmit until null encountered in buffer
       io_Data         pointer to block of data to transmit

   RESULTS
       Error -- if the Write succeded, then io_Error will be null.
           If the Write failed, then the io_Error will be non-zero.

   SEE ALSO
        serial.device/SDCMD_SETPARAMS
```

serial.device Page 12

```
serial.device/OpenDevice                              serial.device/OpenDevice

   NAME
       OpenDevice -- Request an opening of the serial device.

   SYNOPSIS
       error = OpenDevice("serial.device", unit, ioRequest, flags)
       D0                  A0               D0    A1         D1

        BYTE OpenDevice(STRPTR, ULONG, struct IOExtSer *, ULONG);

   FUNCTION
       This is an exec call.  Exec will search for the serial.device, and
       if found, will pass this call on to the device.

       Unless the shared-access bit (bit 5 of io_SerFlags) is set,
       exclusive use is granted and no other access to that unit is
       allowed until the owner closes it.  All the serial-specific fields
       in the ioRequest are initialized to their most recent values (or
       the Preferences default, for the first time open).

       If support of 7-wire handshaking (i.e. RS232-C CTS/RTS protocol)
       is required, use the serial.device/SDCMD_SETPARAMS command.
        This feature should also be specified at inital OpenDevice() time.

   INPUTS
       "serial.device"  - pointer to literal string "serial.device"
       unit             - Must be zero, or a user setable unit number.
                          (This field is used by multiple port controllers)
                         Zero specifies the default serial port.
       ioRequest        - pointer to an ioRequest block of size io_ExtSerSize
                         to be initialized by the serial.device.
                          (see devices/serial.h for the definition)
                          NOTE use of io_SerFlags (see FUNCTION above)
               IMPORTANT: The ioRequest block MUST be of size io_ExtSerSize,
                          and zeroed (with the exeptions as noted)!
       flags            - Must be zero for future compatibility

   RESULTS
       D0         - same as io_Error
       io_Error  - If the Open succeded, then io_Error will be null.
                    If the Open failed, then io_Error will be non-zero.
       io_Device - A pointer to whatever device will handle the calls
                   for this unit.  This pointer may be different depending
                   on what unit is requested.

   BUGS
        If 7-wire handshaking is specified, a timeout "feature" is enabled.
        If the device holds off the computer for more than about 30-60
        seconds, the device will return the write request with the error
        SerErr_TimerErr.  Don't depend on this, however.  If you want a timeout,
        set up the timer.device and wait for either timer, or serial IO to
        complete.

        On open, the serial.device allocates the misc.resource for the
        serial port.  It does not return it until the serial.device is
        expunged from memory.  It should return it when no more openers
        exist.  This code can force a specified device to try and
        expunge.  Of course, if the device is in use nothing will happen:

        #include "exec/types.h"
        #include "exec/execbase.h"
        #include "proto/exec.h"

        void FlushDevice(char *);
```

```
    extern struct ExecBase *SysBase;

    void main()
    {
            FlushDevice("serial.device");   /* or parallel.device */
    }

    /*
     * Attempts to flush the named device out of memory.
     * If it fails, no status is returned; examination of
     * the problem will reveal that information has no
     * valid use after the Permit().
     */
    void FlushDevice(name)
    char  *name;
    {
    struct Device *result;

        Forbid();
        if( result=(struct Device *)FindName(&SysBase->DeviceList,name) )
            RemDevice(result);
        Permit();
    }


SEE ALSO
    serial.device/CloseDevice
    serial.device/SDCMD_SETPARAMS
    devices/serial.h
```

```
serial.device/SDCMD_BREAK                               serial.device/SDCMD_BREAK

   NAME
       Break -- send a break signal over the serial line

   FUNCTION
       This command sends a break signal (serial line held low for an
       extended period) out the serial port. For the built-in port,
       This is accomplished by setting the UARTBRK bit of regisrer ADKCON.

       After a duration (user specifiable via setparams, default 250000
       microseconds) the bit is reset and the signal discontinued.
       If the QUEUEDBRK bit of io_SerFlags is set in the io_Request
       block, the request is placed at the back of the write-request
       queue and executed in turn. If the QUEUEDBRK bit is not set,
       the break is started immediately, control returns to the
       caller, and the timer discontinues the signal after the
       duration is completed. Be aware that calling BREAK may
        affect other commands such as ABORT, FLUSH, STOP, START, etc...

   IO REQUEST
       io_Message      mn_ReplyPort initialized
       io_Device       set by OpenDevice
       io_Unit         set by OpenDevice
       io_Command      SDCMD_BREAK
       io_Flags        set/reset IO_QUICK per above description

   RESULTS
       Error -- if the Break succeded, then Error will be null.
           If the Break failed, then the Error will be non-zero.
```

```
serial.device/SDCMD_QUERY                           serial.device/SDCMD_QUERY

   NAME
       Query -- query serial port/line status

   FUNCTION
       This command return the status of the serial port lines and
       registers. The number of unread bytes in the serial device's
       read buffer is shown in io_Actual.

        The break send & received flags are cleared by a query, and
        whenever a read IORequest is returned with a error
        in io_Error.

   IO REQUEST
       io_Message      mn_ReplyPort initialized
       io_Device       preset by OpenDevice
       io_Unit         preset by OpenDevice
       io_Command      SDCMD_QUERY

   RESULTS
       io_Status        BIT  ACTIVE  FUNCTION

              LSB        0    ---    reserved
                         1    ---    reserved
                         2    high   parallel "sel" on the A1000
                                     On the A500 & A2000, "sel" is also
                                     connected to the serial port's
                                     "Ring Indicator".  Be cautious when
                                     making cables.
                         3    low    Data Set Ready
                         4    low    Clear To Send
                         5    low    Carrier Detect
                         6    low    Ready To Send
                         7    low    Data Terminal Ready
              MSB        8    high   hardware overrun
                         9    high   break sent (most recent output)
                        10    high   break received (as latest input)
                        11    high   transmit x-OFFed
                        12    high   receive x-OFFed
                     13-15    ---    reserved

       io_Actual       set to count of unread input characters

       io_Error -- Query will always succeded.
```

```
serial.device/SDCMD_SETPARAMS                   serial.device/SDCMD_SETPARAMS

   NAME
       SetParams -- change parameters for the serial port

   FUNCTION
       This command allows the caller to change parameters for the
       serial device. Except for xON-xOFF enable/disable, it will
       reject a setparams call if any reads or writes are active
       or pending.
          Note specifically:

    1. Valid input for io_Baud is between 112 and 292000 baud inclusive;
       asynchronous i/o above 32KB (especially on a busy system) may
       be ambitious.
    2. The EOFMODE and QUEUEDBRK bits of io_SerFlags can be set/reset
       in the io_Rqst block without a call to SetParams. The SHARED
       and 7WIRE bits of io_SerFlags can be used in OpenDevice calls.
       ALL OTHER PARAMETERS CAN ONLY BE CHANGED BY THE SetParams
       COMMAND.
    3. RBufLen must be at least 64.  The buffer may be any multiple of
        64 bytes.
    4. If not used, io_ExtFlags MUST be set to zero.
    5. xON-xOFF is by default enabled. The XDISABLED bit is the only
       parameter that can be changed via a SetParams call while the
       device is active. Note that this will return the value
       SerErr_DevBusy in the io_Error field.

        xON/xOFF handshaking is inappropriate for certain binary transfer
        protocalls, such as Xmodem.  The binary data might contain the
        xON (ASCII 17) and xOFF (ASCII 19) characters.

    6. If trying to run MIDI, you should set the RAD_BOOGIE bit of
       io_SerFlags to eliminate unneeded overhead. Specifically, this skips
       checks for parity, x-OFF handling, character lengths other than
       8 bits, and testing for a break signal. Setting RAD_BOOGIE will
       also set the XDISABLED bit.
       Note that writing data (that's already in MIDI format) at MIDI rates
       is easily accomplished. Using this driver alone for MIDI reads may,
       however, may not be reliable, due to MIDI timestamping requirements,
       and possibility of overruns in a busy multitasking and/or display
       intensive environment.
    7. If you select mark or space parity (see io_ExtFlags in serial.h),
       this will cause the SERB_PARTY_ON bit to be set, and the setting
       of SERB_PARTY_ODD to be ignored.
    8. For best results, set the RAD_BOOGIE flag whenever possible.  See
        #6 for details.
    9. Note that at this time parity is *not* calculated for the xON-xOFF
        characters.  If you have a system that is picky about the parity of
        these, you must set your own xON-xOFF characters in io_CtlChar.
   10. 7WIRE (CTS/RTS) handshake is bi-directional.  The external side
        is expected to drop CTS several character times before the external
        buffer is full.  The Amiga will drop RTS several character times
        before the Amiga's buffer is full.


   IO REQUEST
       io_Message      mn_ReplyPort initialized
       io_Device       preset by OpenDevice
       io_Unit         preset by OpenDevice
       io_Command      SDCMD_SETPARAMS (0x0B)
                        NOTE that the following fields are filled in by Open
                       to reflect the serial device's current configuration.
       io_CtlChar      a longword containing byte values for the
                       xON,xOFF,INQ,ACK fields (respectively)
```

```
                (INQ/ACK not used at this time)
    io_RBufLen      length in bytes of input buffer
                     NOTE that any change in buffer size causes the
                    current buffer to be deallocated and a new,
                    correctly sized one to be allocated. Thusly,
                    the CONTENTS OF THE OLD BUFFER ARE LOST.
    io_ExtFlags     additional serial flags (bitdefs in devices/serial.h)
                     mark & space parity may be specified here.
    io_Baud         baud rate for reads AND writes. (See 1 above)
    io_BrkTime      duration of break signal in MICROseconds
    io_TermArray    ASCII descending-ordered 8-byte array of
                    termination characters. If less than 8 chars
                    used, fill out array w/lowest valid value.
                    Terminators are checked only if EOFMODE bit of
                    io_Serflags is set. (e.g. x512F040303030303 )
    io_ReadLen      number of bits in read word (1-8) not including parity
    io_WriteLen     number of bits in write word (1-8) "      "        "
    io_StopBits     number of stop bits (0, 1 or 2)
    io_SerFlags     see devices/serial.h for bit equates, NOTE that x00
                    yields exclusive access, xON/OFF-enabled, no
                    parity checking, 3-wire protocol and TermArray
                    inactive.

RESULTS
    Error -- if the SetParams succeded, then Error will be null.
        If the SetParams failed, then the Error will be non-zero.

SEE ALSO
     exec/OpenDevice
```

TABLE OF CONTENTS

timer.device/--background--
timer.device/AbortIO()
timer.device/AddTime()
timer.device/CmpTime()
timer.device/GetSysTime()
timer.device/ReadEClock()
timer.device/SubTime()
timer.device/TR_ADDREQUEST
timer.device/TR_GETSYSTIME
timer.device/TR_SETSYSTIME

```
timer.device/--background--                          timer.device/--background--

   TIMER REQUEST
        A time request is a non standard IO Request.  It has an IORequest
        followed by a timeval structure or an eclockval structure.

   TIMEVAL
        A timeval structure consists of two longwords.  The first is
        the number of seconds, the latter is the fractional number
        of microseconds.  The microseconds must always be "normalized"
        e.g. the longword must be between 0 and one million.

   ECLOCKVAL
        A eclockval structure consists of two longwords.  The first is
        the high order 32 bits of a 64 bit number and the second is the
        the low order 32 bits.  The 64 bit number is a count of "E" clock
        ticks.  The "E" clock frequency is related to the master clock
        frequency of the machine and can be determined by calling the
        ReadEClock() library like call.

   UNITS
        The timer contains five units -- two designed to accuratly measure
        short intervals, one that has little system overhead and is very
        stable over time, and two that work like an alarm clock.

   UNIT_MICROHZ
        This unit uses the programmable timers in the 8520s to keep track
        of its time.  It has precision down to about 2 microseconds, but
        will drift as system load increases.  The accuracy of this unit
        is the same as that of the master clock of the machine.  This unit
        uses a timeval in its timerequest.

   UNIT_VBLANK
        This unit uses a strobe from the power supply to keep track of its
        time or the "E" clock on machines without power supply strobes.
        It is very stable over time, but only has a resolution of that of
        the vertical blank interrupt.  This unit is very cheap to use, and
        should be used by those who are waiting for long periods of time
        (typically 1/2 second or more).  This unit uses a timeval in its
        timerequest.

   UNIT_ECLOCK
        This unit is exacly the same as UNIT_MICROHZ except that it uses
        an eclockval instead of a timeval in its timerequest.

   UNIT_WAITUNTIL
        This unit waits until the systime is greater than or equal to the
        time in the timeval in the timerequest.  This unit has the same
        resolution and accuracy as that of UNIT_VBLANK.

   UNIT_WAITECLOCK
        This unit waits until the E-Clock value as returned by ReadEClock()
        is greater than or equal to the eclockval in the timerequest. This
        unit has the same resolution and accuracy as that of UNIT_ECLOCK.

   LIBRARY
        In addition to the normal device calls, the timer also supports
        several direct, library like calls.

   BUGS
        In the V1.2/V1.3 release, the timer device has problems with
        very short time requests.  When one of these is made, other
        timer requests may be finished inaccurately.  A side effect
        is that AmigaDOS requests such as "Delay(0);" or
        "WaitForChar(x,0);" are unreliable.
```

timer.device Page 3

```
timer.device/AbortIO()                                 timer.device/AbortIO()

   NAME
       AbortIO -- Remove an existing timer request.

   SYNOPSIS
       error = AbortIO( timerequest )
       D0               A1

       LONG AbortIO( struct timerequest * );

   FUNCTION
       This is an exec.library call.

       This routine removes a timerquest from the timer.  It runs in
       the context of the caller.

   INPUTS
       timerequest - the timer request to be aborted

   RETURNS
       0  if the request was aborted, io_Error will also be set to
          IOERR_ABORTED.
       -1 otherwise

   NOTES
       This function may be called from interrupts.

   SEE ALSO
       exec.library/AbortIO()

   BUGS
```

timer.device Page 4

```
timer.device/AddTime()                                 timer.device/AddTime()

   NAME
       AddTime -- Add one time request to another.

   SYNOPSIS
       AddTime( Dest, Source )
                A0    A1

       void AddTime( struct timeval *, struct timeval *);

   FUNCTION
       This routine adds one timeval structure to another.  The
       results are stored in the destination (Dest + Source -> Dest)

       A0 and A1 will be left unchanged

   INPUTS
       Dest, Source -- pointers to timeval structures.

   NOTES
       This function may be called from interrupts.

   SEE ALSO
       timer.device/CmpTime(),
       timer.device/SubTime()

   BUGS
```

```
timer.device/CmpTime()                                  timer.device/CmpTime()

   NAME
        CmpTime -- Compare two timeval structures.

   SYNOPSIS
        result = CmpTime( Dest, Source )
        D0                A0    A1

        LONG CmpTime( struct timeval *, struct timeval *);

   FUNCTION
        This routine compares timeval structures

        A0 and A1 will be left unchanged

   INPUTS
        Dest, Source -- pointers to timeval structures.

   RESULTS
        result will be   0 if Dest has same time as source
                        -1 if Dest has more time than source
                        +1 if Dest has less time than source

   NOTES
        This function may be called from interrupts.

   SEE ALSO
        timer.device/AddTime(),
        timer.device/SubTime()

   BUGS
        Older version of this document had the sense of the return
        codes wrong; the code hasn't changed but the document has.
```

```
timer.device/GetSysTime()                                 timer.device/GetSysTime()

   NAME
        GetSysTime -- Get the system time. (V36)

   SYNOPSIS
        GetSysTime( Dest )
                    A0

        void GetSysTime( struct timeval * );

   FUNCTION
        Ask the system what time it is.  The system time starts off at
        zero at power on, but may be initialized via the TR_SETSYSTIME
        timer.device command.

        System time is monotonocally increasing and guarenteed to be
        unique (except when the system time is set back).

        A0 will be left unchanged.

        This function is less expensive to use than the TR_GETSYSTIME
        IORequest.

   INPUTS
        Dest -- pointer to a timeval structure to hold the system time.

   RESULTS
        Dest -- the timeval structure will contain the system time.

   NOTES
        This function may be called from interrupts.

   SEE ALSO
        timer.device/TR_GETSYSTIME,
        timer.device/TR_SETSYSTIME,

   BUGS
```

timer.device Page 7

```
timer.device/ReadEClock()                              timer.device/ReadEClock()

   NAME
        ReadEClock -- Get the current value of the E-Clock. (V36)

   SYNOPSIS
        E_Freq = ReadEClock( Dest )
        D0                   A0

        ULONG ReadEClock ( struct EClockVal * );

   FUNCTION
        This routine calculates the current 64 bit value of the E-Clock
        and stores it in the destination EClockVal structure. The count
        rate of the E-Clock is also returned.

        A0 will be left unchanged

        This is a low overhead function designed so that very short
        intervals may be timed.

   INPUTS
        Dest -- pointer to an EClockVal structure.

   RETURNS
        Dest -- the EClockVal structure will contain the E-Clock time
        E_Freq -- The count rate of the E-Clock (tics/sec).

   NOTES
        This function may be called from interrupts.

   SEE ALSO

   BUGS
```

timer.device Page 8

```
timer.device/SubTime()                                       timer.device/SubTime()

   NAME
        SubTime -- Subtract one time request from another.

   SYNOPSIS
        SubTime( Dest, Source )
                 A0    A1

        void SubTime( struct timeval *, struct timeval *);

   FUNCTION
        This routine subtracts one timeval structure from another.  The
        results are stored in the destination (Dest - Source -> Dest)

        A0 and A1 will be left unchanged

   INPUTS
        Dest, Source -- pointers to timeval structures.

   NOTES
        This function may be called from interrupts.

   SEE ALSO
        timer.device/AddTime(),
        timer.device/CmpTime()

   BUGS
```

```
timer.device/TR_ADDREQUEST                          timer.device/TR_ADDREQUEST

   NAME
       TR_ADDREQUEST -- Submit a request to wait a period of time.

   FUNCTION
       Ask the timer to wait a specified amount of time before
       replying the timerequest.

       The message may be forced to finish early with an
       AbortIO()/WaitIO() pair.

   TIMER REQUEST
       io_Message       mn_ReplyPort initialized
       io_Device        preset by timer in OpenDevice
       io_Unit          preset by timer in OpenDevice
       io_Command       TR_ADDREQUEST
       io_Flags         IOF_QUICK permitted (but ignored)
       tr_time          a timeval structure specifying how long the
                            device will wait before replying

   RESULTS
       tr_time          will be zeroed

   NOTES
       This function may be called from interrupts.

       Previous to 2.0, the tr_time field was documented as containing
       junk when the timerequest was returned.

   SEE ALSO
       timer.device/AbortIO(),
       timer.device/TimeDelay(),

   BUGS
```

```
timer.device/TR_GETSYSTIME                          timer.device/TR_GETSYSTIME

   NAME
       TR_GETSYSTIME -- get the system time.

   FUNCTION
       Ask the system what time it is.  The system time starts off at
       zero at power on, but may be initialized via the TR_SETSYSTIME
       call.

       System time is monotonically increasing, and guaranteed to be
       unique (except when the system time is set backwards).

   TIMER REQUEST
       io_Message       mn_ReplyPort initialized
       io_Device        preset by timer in OpenDevice
       io_Unit          preset by timer in OpenDevice
       io_Command       TR_GETSYSTIME
       io_Flags         IOF_QUICK permitted

   RESULTS
       tr_time          a timeval structure with the current system
                            time

   NOTES
       This function may be called from interrupts.

   SEE ALSO
       timer.device/TR_SETSYSTIME,
       timer.device/GetSysTime(),

   BUGS
```

```
timer.device/TR_SETSYSTIME                              timer.device/TR_SETSYSTIME

   NAME
        TR_SETSYSTIME -- Set the system time.

   FUNCTION
        Set the system idea of what time it is.  The system starts out
        at time "zero" so it is safe to set it forward to the real
        time.  However, care should be taken when setting the time
        backwards.  System time is generally expected to monotonically
        increasing.

   TIMER REQUEST
        io_Message      mn_ReplyPort initialized
        io_Device       preset by timer in OpenDevice
        io_Unit         preset by timer in OpenDevice
        io_Command      TR_GETSYSTIME
        io_Flags        IOF_QUICK permitted
        tr_time         a timeval structure with the current system
                            time

   RESULTS
        tr_time         will contain junk

   NOTES
        This function may be called from interrupts.

   SEE ALSO
        timer.device/TR_GETSYSTIME,
        timer.device/GetSysTime(),

   BUGS
```

trackdisk.device Page 1

TABLE OF CONTENTS

trackdisk.device/CMD_CLEAR
trackdisk.device/CMD_READ
trackdisk.device/CMD_UPDATE
trackdisk.device/CMD_WRITE
trackdisk.device/TD_ADDCHANGEINT
trackdisk.device/TD_CHANGENUM
trackdisk.device/TD_CHANGESTATE
trackdisk.device/TD_EJECT
trackdisk.device/TD_FORMAT
trackdisk.device/TD_GETDRIVETYPE
trackdisk.device/TD_GETGEOMETRY
trackdisk.device/TD_GETNUMTRACKS
trackdisk.device/TD_MOTOR
trackdisk.device/TD_PROTSTATUS
trackdisk.device/TD_RAWREAD
trackdisk.device/TD_RAWWRITE
trackdisk.device/TD_REMCHANGEINT
trackdisk.device/TD_SEEK

trackdisk.device Page 2

```
trackdisk.device/CMD_CLEAR                          trackdisk.device/CMD_CLEAR

   NAME
        CMD_CLEAR/ETD_CLEAR -- mark the track buffer as containing invalid
                               data.

   FUNCTION
        These commands mark the track buffer as invalid, forcing a
        reread of the disk on the next operation. ETD_UPDATE or CMD_UPDATE
        would be used to force data out to the disk before turning the motor
        off. ETD_CLEAR or CMD_CLEAR are usually used after having locked out
        the trackdisk.device via the use of the disk resource, when you
        wish to prevent the track from being updated, or when you wish to
        force the track to be re-read. ETD_CLEAR or CMD_CLEAR will not do an
        update, nor will an update command do a clear.

   IO REQUEST INPUT
        io_Device       preset by the call to OpenDevice()
        io_Unit         preset by the call to OpenDevice()
        io_Command      CMD_CLEAR or ETD_CLEAR
        io_Flags        0 or IOF_QUICK
        iotd_Count      (ETD_CLEAR only) maximum allowable change counter
                        value.

   IO REQUEST RESULT
        io_Error - 0 for success, or am error code as defined in
                <devices/trackdisk.h>

   SEE ALSO
        CMD_WRITE, CMD_UPDATE
```

trackdisk.device Page 3

```
trackdisk.device/CMD_READ                             trackdisk.device/CMD_READ

   NAME
       CMD_READ/ETD_READ -- read sectors of data from a disk.

   FUNCTION
       These commands transfer data from the track buffer to a supplied
       buffer. If the desired sector is already in the track buffer, no disk
       activity is initiated. If the desired sector is not in the buffer, the
       track containing that sector is automatically read in. If the data in
       the current track buffer has been modified, it is written out to the
       disk before a new track is read. ETD_READ will read the sector label
       area if the iotd_SecLabel is non-NULL.

   IO REQUEST INPUT
       io_Device       preset by the call to OpenDevice()
       io_Unit         preset by the call to OpenDevice()
       io_Command      CMD_READ or ETD_READ
       io_Flags        0 or IOF_QUICK
       io_Data         pointer to the buffer where the data should be put
       io_Length       number of bytes to read, must be a multiple of
                       TD_SECTOR.
       io_Offset       byte offset from the start of the disk describing
                       where to read data from, must be a multiple of
                       TD_SECTOR.
       iotd_Count      (ETD_READ only) maximum allowable change counter
                       value.
       iotd_SecLabel   (ETD_READ only) NULL or sector label buffer pointer.
                       If provided, the buffer must be a multiple of
                       TD_LABELSIZE.

   IO REQUEST RESULT
       io_Error - 0 for success, or am error code as defined in
                  <devices/trackdisk.h>
   NOTES
       Under versions of Kickstart earlier than V36, the io_Data had to
       point to a buffer in chip memory. This restriction is no longer
       present as of Kickstart V36 and beyond.

   SEE ALSO
       CMD_WRITE
```

trackdisk.device Page 4

```
trackdisk.device/CMD_UPDATE                         trackdisk.device/CMD_UPDATE

   NAME
       CMD_UPDATE/ETD_UPDATE -- write out the track buffer if it is dirty.

   FUNCTION
       The trackdisk device does not write data sectors unless it is
       necessary (you request that a different track be used) or until the
       user requests that an update be performed. This improves system speed
       by caching disk operations. These commands ensure that any
       buffered data is flushed out to the disk. If the track buffer has not
       been changed since the track was read in, these commands do nothing.
       ETD_UPDATE command checks for diskchange.

   IO REQUEST INPUT
       io_Device       preset by the call to OpenDevice()
       io_Unit         preset by the call to OpenDevice()
       io_Command      CMD_UPDATE or ETD_UPDATE
       io_Flags        0 or IOF_QUICK
       iotd_Count      (ETD_UPDATE only) maximum allowable change counter
                       value.

   IO REQUEST RESULT
       io_Error - 0 for success, or am error code as defined in
                  <devices/trackdisk.h>

   SEE ALSO
       CMD_WRITE
```

trackdisk.device Page 5

```
trackdisk.device/CMD_WRITE                        trackdisk.device/CMD_WRITE

   NAME
       CMD_WRITE/ETD_WRITE -- write sectors of data to a disk.

   FUNCTION
       These commands transfer data from a supplied buffer to the track
       buffer. If the track that contains this sector is already in the track
       buffer, no disk activity is initiated. If the desired sector is not in
       the buffer, the track containing that sector is automatically read in.
       If the data in the current track buffer has been modified, it is
       written out to the disk before the new track is read in for
       modification. ETD_WRITE will write the sector label area if
       iotd_SecLabel is non-NULL.

   IO REQUEST INPUT
       io_Device       preset by the call to OpenDevice()
       io_Unit         preset by the call to OpenDevice()
       io_Command      CMD_WRITE or ETD_WRITE
       io_Flags        0 or IOF_QUICK
       io_Data         pointer to the buffer where the data should be put
       io_Length       number of bytes to write, must be a multiple of
                       TD_SECTOR.
       io_Offset       byte offset from the start of the disk describing
                       where to write data to, must be a multiple of
                       TD_SECTOR.
       iotd_Count      (ETD_WRITE only) maximum allowable change counter
                       value.
       iotd_SecLabel   (ETD_WRITE only) NULL or sector label buffer pointer.
                       If provided, the buffer must be a multiple of
                       TD_LABELSIZE.

   IO REQUEST RESULT
       io_Error - 0 for success, or am error code as defined in
                  <devices/trackdisk.h>

   NOTES
       Under versions of Kickstart earlier than V36, the io_Data had to
       point to a buffer in chip memory. This restriction is no longer
       present as of Kickstart V36 and beyond.

   SEE ALSO
       CMD_READ, TD_FORMAT
```

trackdisk.device Page 6

```
trackdisk.device/TD_ADDCHANGEINT                  trackdisk.device/TD_ADDCHANGEINT

   NAME
       TD_ADDCHANGEINT -- add a disk change software interrupt handler.

   FUNCTION
       This command lets you add a software interrupt handler to the
       disk device that gets invoked whenever a disk insertion or removal
       occurs.

       You must pass in a properly initialized Exec Interrupt structure
       and be prepared to deal with disk insertions/removals
       immediately. From within the interrupt handler, you may only call the
       status commands that can use IOF_QUICK.

       To set up the handler, an Interrupt structure must be initialized.
       This structure is supplied as the io_Data to the TD_ADDCHANGEINT
       command. The handler then gets linked into the handler chain and
       gets invoked whenever a disk change happens. You must eventually
       remove the handler before you exit.

       This command only returns when the handler is removed. That is,
       the device holds onto the IO request until the TD_REMCHANGEINT command
       is executed with that same IO request. Hence, you must use SendIO()
       with this command.

   IO REQUEST INPUT
       io_Device       preset by the call to OpenDevice()
       io_Unit         preset by the call to OpenDevice()
       io_Command      TD_ADDCHANGEINT
       io_Flags        0
       io_Length       sizeof(struct Interrupt)
       io_Data         pointer to Interrupt structure

   IO REQUEST RESULT
       io_Error - 0 for success, or am error code as defined in
                  <devices/trackdisk.h>

   SEE ALSO
       TD_REMCHANGEINT, <devices/trackdisk.h>, <exec/interrupts.h>,
       exec.library/Cause()
```

trackdisk.device

```
trackdisk.device/TD_CHANGENUM                     trackdisk.device/TD_CHANGENUM

   NAME
       TD_CHANGENUM -- return the current value of the disk-change counter.

   FUNCTION
       This command returns the current value of the disk-change counter (as
       used by the enhanced commands). The disk change counter is incremented
       each time a disk is inserted or removed from the trackdisk unit.

   IO REQUEST INPUT
       io_Device       preset by the call to OpenDevice()
       io_Unit         preset by the call to OpenDevice()
       io_Command      TD_CHANGENUM
       io_Flags        0 or IOF_QUICK

   IO REQUEST RESULT
       io_Error - 0 for success, or am error code as defined in
                  <devices/trackdisk.h>
       io_Actual - if io_Error is 0, this contains the current value of the
                   disk-change counter.
```

trackdisk.device

```
trackdisk.device/TD_CHANGESTATE                 trackdisk.device/TD_CHANGESTATE

   NAME
       TD_CHANGESTATE -- check if a disk is currently in a drive.

   FUNCTION
       This command checks to see if there is currently a disk in a drive.

   IO REQUEST INPUT
       io_Device       preset by the call to OpenDevice()
       io_Unit         preset by the call to OpenDevice()
       io_Command      TD_CHANGESTATE
       io_Flags        0 or IOF_QUICK

   IO REQUEST RESULT
       io_Error - 0 for success, or am error code as defined in
                  <devices/trackdisk.h>
       io_Actual - if io_Error is 0, this tells you whether a disk is in
                   the drive. 0 means there is a disk, while anything else
                   indicates there is no disk.
```

trackdisk.device

Page 9

```
trackdisk.device/TD_EJECT                              trackdisk.device/TD_EJECT

   NAME
       TD_EJECT -- eject the disk in the drive, if possible.

   FUNCTION
        This command causes the drive to attempt to eject the disk in
        it, if any.  Note that the current trackdisk.device does not
        implement this command, but it might in the future, and other
        trackdisk-compatible drivers may implement this command.

   IO REQUEST INPUT
        io_Device       preset by the call to OpenDevice()
        io_Unit         preset by the call to OpenDevice()
        io_Command      TD_EJECT
        io_Flags        0 or IOF_QUICK

   IO REQUEST RESULT
        io_Error - 0 for success, or am error code as defined in
                   <devices/trackdisk.h>
```

trackdisk.device

Page 10

```
trackdisk.device/TD_FORMAT                             trackdisk.device/TD_FORMAT

   NAME
        TD_FORMAT/ETD_FORMAT -- format a track on a disk.

   FUNCTION
        These commands are used to write data to a track that either
        has not yet been formatted or has had a hard error on a standard write
        command. TD_FORMAT completely ignores all data currently on a track and
        does not check for disk change before performing the command. The
        io_Data field must point to at least one track worth of data. The
        io_Offset field must be track aligned, and the io_Length field must be
        in units of track length (that is, NUMSEC*TD_SECTOR).

        The device will format the requested tracks, filling each sector with
        the contents of the buffer pointed to by io_Data. You
        should do a read pass to verify the data.

        If you have a hard write error during a normal write, you may find it
        possible to use the TD_FORMAT command to reformat the track as part of
        your error recovery process. ETD_FORMAT will write the sector label
        area if iotd_SecLabel is non-NULL.

   IO REQUEST INPUT
        io_Device       preset by the call to OpenDevice()
        io_Unit         preset by the call to OpenDevice()
        io_Command      TD_FORMAT or ETD_FORMAT
        io_Flags        0 or IOF_QUICK
        io_Data         points to a buffer containing the data to write to the
                        track, must be at least as large as io_Length.
        io_Length       number of bytes to format, must be a multiple of
                        (TD_SECTORS * NUMSEC).
        io_Offset       byte offset from the start of the disk for the track to
                        format, must be a multiple of (TD_SECTORS * NUMSEC).
        iotd_Count      (ETD_FORMAT only) maximum allowable change counter
                        value.
        iotd_SecLabel   (ETD_FORMAT only) NULL or sector label buffer pointer.
                        If provided, the buffer must be a multiple of
                        (TD_LABELSIZE * NUMSEC).

   IO REQUEST RESULT
        io_Error - 0 for success, or am error code as defined in
                   <devices/trackdisk.h>

   NOTES
        Under versions of Kickstart earlier than V36, the io_Data had to
        point to a buffer in chip memory. This restriction is no longer
        present as of Kickstart V36 and beyond.

   SEE ALSO
        CMD_WRITE, TD_RAWWRITE
```

trackdisk.device Page 11

```
trackdisk.device/TD_GETDRIVETYPE               trackdisk.device/TD_GETDRIVETYPE

   NAME
        TD_GETDRIVETYPE -- return the type of disk drive for the unit that was
                           opened.

   FUNCTION
        This command returns the type of the disk drive to the user.
        This number will be a small integer and will come from the set of
        DRIVEXXX constants defined in <devices/trackdisk.h>.

        The only way you can actually use this command is if the trackdisk
        device understands the drive type of the hardware that is plugged in.
        This is because the OpenDevice() call will fail if the trackdisk device
        does not understand the drive type. To find raw drive identifiers see
        the disk.resource's DR_GETUNITID entry point.

   IO REQUEST INPUT
        io_Device       preset by the call to OpenDevice()
        io_Unit         preset by the call to OpenDevice()
        io_Command      TD_GETDRIVETYPE
        io_Flags        0 or IOF_QUICK

   IO REQUEST RESULT
        io_Error - 0 for success, or am error code as defined in
                   <devices/trackdisk.h>
        io_Actual - if io_Error is 0 this contains the drive type connected to
                    this unit.

   SEE ALSO
        TD_GETNUMTRACKS, <devices/trackdisk.h>
```

trackdisk.device Page 12

```
trackdisk.device/TD_GETGEOMETRY                 trackdisk.device/TD_GETGEOMETRY

   NAME
        TD_GETGEOMETRY -- return the geometry of the drive.

   FUNCTION
        This command returns a full set of information about the
        layout of the drive. The information is returned in the
        DriveGeometry structure pointed to by io_Data.

   IO REQUEST INPUT
        io_Device       preset by the call to OpenDevice()
        io_Unit         preset by the call to OpenDevice()
        io_Command      TD_GETGEOMETRY
        io_Flags        0 or IOF_QUICK
        io_Data         Pointer to a DriveGeometry structure
        io_Length       sizeof(struct DriveGeometry)

   IO REQUEST RESULT
        io_Error - 0 for success, or am error code as defined in
                   <devices/trackdisk.h>

   NOTE
        This information may change when a disk in inserted when
        certain hardware is present.

   SEE ALSO
        TD_GETDRIVETYPE, TD_GETNUMTRACKS
```

trackdisk.device

```
trackdisk.device/TD_GETNUMTRACKS            trackdisk.device/TD_GETNUMTRACKS

   NAME
       TD_GETNUMTRACKS -- return the number of tracks for the type of disk
                          drive for the unit that was opened.

   FUNCTION
       This command returns the number of tracks that are available
       on the disk unit.

   IO REQUEST INPUT
       io_Device       preset by the call to OpenDevice()
       io_Unit         preset by the call to OpenDevice()
       io_Command      TD_GETNUMTRACKS
       io_Flags        0 or IOF_QUICK

   IO REQUEST RESULT
       io_Error - 0 for success, or am error code as defined in
                  <devices/trackdisk.h>
       io_Actual - if io_Error is 0 this contains the drive type connected to
                  this unit.

   SEE ALSO
       TD_GETDRIVETYPE
```

---

trackdisk.device

```
trackdisk.device/TD_MOTOR                                trackdisk.device/TD_MOTOR

   NAME
       TD_MOTOR/ETD_MOTOR -- control the on/off state of a drive motor.

   FUNCTION
       This command gives control over the disk motor. The motor may be
       turned on or off. When it is on, the drive light automatically turns
       on as well.

       If the motor is just being turned on, the device will delay the
       proper amount of time to allow the drive to come up to speed. Normally,
       turning the drive on is not necessary, the device does this
       automatically if it receives a request when the motor is off. However,
       turning the motor off is the programmer's responsibility.

       In addition, the standard instructions to the user are that it is safe
       to remove a disk from a drive if and only if the motor is off (that is,
       if the disk light is off).

   IO REQUEST INPUT
       io_Device       preset by the call to OpenDevice()
       io_Unit         preset by the call to OpenDevice()
       io_Command      TD_MOTOR or ETD_MOTOR
       io_Flags        0 or IOF_QUICK
       io_Length       the requested state of the motor, 0 to turn the motor
                       off, and 1 to turn the motor on.
       iotd_Count      (ETD_MOTOR only) maximum allowable change counter
                       value.

   IO REQUEST RESULT
       io_Error - 0 for success, or am error code as defined in
                  <devices/trackdisk.h>
       io_Actual - if io_Error is 0 this contains the previous state of the
                   drive motor.
```

```
trackdisk.device/TD_PROTSTATUS                        trackdisk.device/TD_PROTSTATUS

   NAME
        TD_PROTSTATUS -- return whether the current disk is write-protected.

   FUNCTION
        This command is used to determine whether the current disk is
        write-protected.

   IO REQUEST INPUT
        io_Device       preset by the call to OpenDevice()
        io_Unit         preset by the call to OpenDevice()
        io_Command      TD_PROTSTATUS
        io_Flags        0 or IOF_QUICK

   IO REQUEST RESULT
        io_Error - 0 for success, or am error code as defined in
                   <devices/trackdisk.h>
        io_Actual - if io_Error is 0, this tells you whether the disk in the
                   drive is write-protected. 0 means the disk is NOT write-
                   protected, while any other value indicates it is.
```

```
trackdisk.device/TD_RAWREAD                                trackdisk.device/TD_RAWREAD

   NAME
        TD_RAWREAD/ETD_RAWREAD -- read raw data from the disk.

   FUNCTION
        These commands read a track of raw data from disk and deposits it in
        the provided buffer. The data is taken straight from the disk with
        no processing done on it. It will appear exactly as the bits come out
        off the disk, hopefully in some legal MFM format.

        This interface is intended for sophisticated programmers only.
        Commodore-Amiga reserves the right to make enhancements to the disk
        format in the future. We will provide compatibility via the
        CMD_READ and ETD_READ commands, anyone using TD_RAWREAD is
        bypassing this upwards compatibility, and may thus stop working.

   IO REQUEST INPUT
        io_Device       preset by the call to OpenDevice()
        io_Unit         preset by the call to OpenDevice()
        io_Command      TD_RAWREAD or ETD_RAWREAD.
        io_Flags        if the IOTDB_INDEXSYNC bit is set then the driver
                        will make a best effort attempt to start reading
                        from the index mark.  Note that there will be at
                        least some delay, and perhaps a great deal of delay
                        (for example if interrupts have been disabled).
        io_Length       Length of buffer in bytes, with a maximum of 32768
                        bytes.
        io_Data         Pointer to CHIP memory buffer where raw track data is
                        to be deposited.
        io_Offset       The number of the track to read in.
        iotd_Count      (ETD_RAWREAD only) maximum allowable change counter
                        value.

   IO REQUEST RESULT
        io_Error - 0 for success, or am error code as defined in
                   <devices/trackdisk.h>

   NOTES
        The track buffer provided MUST be in CHIP memory

        There is a delay between the index pulse and the start of bits
        coming in from the drive (e.g. dma started). This delay
        is in the range of 135-200 microseconds. This delay breaks
        down as follows: 55 microsecs is software interrupt overhead
        (this is the time from interrupt to the write of the DSKLEN
        register). 66 microsecs is one horizontal line delay (remember
        that disk IO is synchronized with agnus' display fetches).
        The last variable (0-65 microsecs) is an additional scan line
        since DSKLEN is poked anywhere in the horizontal line. This leaves
        15 microsecs unaccounted for...  Sigh.

        In short, You will almost never get bits within the first 135
        microseconds of the index pulse, and may not get it until 200
        microseconds. At 4 microsecs/bit, this works out to be between
        4 and 7 bytes of user data of delay.

   BUGS
        This command does not work reliably under versions of Kickstart
        earlier than V36, especially on systems with 1 floppy drive.

   SEE ALSO
        TD_RAWWRITE
```

trackdisk.device Page 17

```
trackdisk.device/TD_RAWWRITE                          trackdisk.device/TD_RAWWRITE

   NAME
        TD_RAWWRITE/ETD_RAWWRITE -- write raw data to the disk.

   FUNCTION
        This command writes a track of raw data from the provided buffer to
        the specified track on disk. The data is copied straight to the disk
        with no processing done on it. It will appear exactly on the disk as
        it is in the memory buffer, hopefully in a legal MFM format.

        This interface is intended for sophisticated programmers only.
        Commodore-Amiga reserves the right to make enhancements to the disk
        format in the future. We will provide compatibility via the
        CMD_WRITE and ETD_WRITE commands, anyone using TD_RAWWRITE is
        bypassing this upwards compatibility, and may thus stop working.

   IO REQUEST INPUT
        io_Device       preset by the call to OpenDevice()
        io_Unit         preset by the call to OpenDevice()
        io_Command      TD_RAWWRITE or ETD_RAWWRITE.
        io_Flags        if the IOTDB_INDEXSYNC bit is set then the driver
                        will make a best effort attempt to start writing
                        from the index mark.  Note that there will be at
                        least some delay, and perhaps a great deal of delay
                        (for example if interrupts have been disabled).
        io_Length       Length of buffer in bytes, with a maximum of 32768
                        bytes.
        io_Data         Pointer to CHIP memory buffer where raw track data is
                        to be taken.
        io_Offset       The number of the track to write to.
        iotd_Count      (ETD_RAWWRITE only) maximum allowable change counter
                        value.

   IO REQUEST RESULT
        io_Error - 0 for success, or am error code as defined in
                   <devices/trackdisk.h>

   NOTES
        The track buffer provided MUST be in CHIP memory

        There is a delay between the index pulse and the start of bits
        going out to the driver (e.g. write gate enabled). This delay
        is in the range of 135-200 microseconds. This delay breaks
        down as follows: 55 microsecs is software interrupt overhead
        (this is the time from interrupt to the write of the DSKLEN
        register). 66 microsecs is one horizontal line delay (remember
        that disk IO is synchronized with agnus' display fetches).
        The last variable (0-65 microsecs) is an additional scan line
        since DSKLEN is poked anywhere in the horizontal line. This leaves
        15 microsecs unaccounted for...  Sigh.

        In short, You will almost never get bits within the first 135
        microseconds of the index pulse, and may not get it until 200
        microseconds. At 4 microsecs/bit, this works out to be between
        4 and 7 bytes of user data of delay.

   BUGS
        This command does not work reliably under versions of Kickstart
        earlier than V36, especially on systems with 1 floppy drive.

   SEE ALSO
        TD_RAWREAD
```

trackdisk.device Page 18

```
trackdisk.device/TD_REMCHANGEINT                  trackdisk.device/TD_REMCHANGEINT

   NAME
        TD_REMCHANGEINT -- remove a disk change software interrupt handler.

   FUNCTION
        This command removes a disk change software interrupt added
        by a previous use of TD_ADDCHANGEINT.

   IO REQUEST INPUT
        The same IO request used for TD_ADDCHANGEINT.

        io_Device       preset by the call to OpenDevice()
        io_Unit         preset by the call tc OpenDevice()
        io_Command      TD_REMCHANGEINT
        io_Flags        0
        io_Length       sizeof(struct Interrupt)
        io_Data         pointer to Interrupt structure

   IO REQUEST RESULT
        io_Error - 0 for success, or am error code as defined in
                   <devices/trackdisk.h>

   BUGS
        This command did not function properly under versions of Kickstart
        earlier than V36. A valid workaround under these older versions of
        Kickstart is:

                Forbid();
                Remove(ioRequest);
                Permit();

        Do not use this workaround in versions of Kickstart >= V36, use
        TD_REMCHANGEINT instead (for future compatibility with V38+).

   SEE ALSO
        TD_ADDCHANGEINT, <devices/trackdisk.h>
```

```
trackdisk.device/TD_SEEK                             trackdisk.device/TD_SEEK

   NAME
        TD_SEEK/ETD_SEEK -- control positioning of the drive heads.

   FUNCTION
        These commands are currently provided for internal diagnostics,
        disk repair, and head cleaning only.

        TD_SEEK and ETD_SEEK move the drive heads to the track specified. The
        io_Offset field should be set to the (byte) offset to which the seek is
        to occur. TD_SEEK and ETD_SEEK do not verify their position until the
        next read. That is, they only move the heads; they do not actually read
        any data.

   IO REQUEST INPUT
        io_Device       preset by the call to OpenDevice()
        io_Unit         preset by the call to OpenDevice()
        io_Command      TD_SEEK or ETD_SEEK
        io_Flags        0 or IOF_QUICK
        io_Offset       byte offset from the start of the disk describing
                        where to move the head to.
        iotd_Count      (ETD_SEEK only) maximum allowable change counter
                        value.

   IO REQUEST RESULT
        io_Error - 0 for success, or am error code as defined in
                   <devices/trackdisk.h>
```

Resource Autodocs

This section contains summaries for the Amiga's system resource routines. Resources contain low-level hardware control functions that arbitrate control of chip registers, interrupts and other low-level hardware. Resources should not be used unless your program has special, high-performance requirements that cannot be met through the conventional function interface provided by the Amiga's library and device functions.

WARNING: Under the multitasking operating system, user-level tasks are generally *not* allowed to directly use the hardware features. If your program requires direct hardware access, resources provide a way of asking for ownership of the needed hardware components. Indiscriminate hardware meddling will cause problems the next time the hardware or operating system is upgraded.

There are currently seven standard resources in the Amiga system:

- ☐ battclock.resource grants access to the Amiga's clock chip. Not all Amiga's have the clock chip, so use this with care.
- ☐ battmem.resource grants access to non-volatile RAM. Not all Amiga's have non-volatile RAM, so use this with care.
- ☐ cia.resource grants access to specific bits and individual interrupts for each of the 8520 CIA chips (Complex Interface Adapters). There are two cia resources: ciaa.resource and ciab.resource, which correspond to the odd and even 8520 CIA chips.
- ☐ disk.resource grants temporary exclusive access to the disk hardware (one for each of the four possible disk/MFM units).
- ☐ FileSystem.resource provides a list of filesystems available for use.

☐ misc.resource grants exclusive access to functional blocks of chip registers. At this time definitions have been made for the serial and parallel hardware. When a task owns the misc.resource for a port, it has control over that port's associated hardware.

☐ potgo.resource manages the bits of the POTGO (write-only) and POTINP (read-only) registers. These custom chip registers control the proportional input pins on the game controller ports. The pins may also be used for digital input and output. Intuition uses port 1 for reading the right and (optional) middle mouse buttons.

See the Amiga *Hardware Reference Manual* for more information on the actual hardware that each resource controls. This section covers only the arbitration provided in the Amiga's multitasking system.

WARNING: Resources are just one step above direct hardware manipulation. You are advised to try the higher level device and library approach before resorting to the hardware.

```
* Assembly language fragment that grabs one of the two groups of serial
* port bits (using the misc.resource).  If it is successful at obtaining
* the resource, it will hang on to it forever, and never return.
*
* This example must be linked with amiga.lib
*
             INCLUDE "exec/types.i"
             INCLUDE "resources/misc.i"

_AbsExecBase equ 4
JSRLIB       MACRO
             XREF    _LVO\1
             JSR     _LVO\1(A6)
             ENDM

             move.l _AbsExecBase,a6
             lea.l         MiscName(pc),a1
             JSRLIB OpenResource
             tst.l  d0
             beq.s  no_open
             move.l d0,a6       ;resource base in A6
;
; We now have a pointer to a resource.
; Call one of its library-like vectors.
;
             move.l #MR_SERIALPORT,d0   ;We want these bits
             lea.l  MyName(pc),a1       ;This is our name
             jsr    MR_ALLOCMISCRESOURCE(a6)
             tst.l  d0
             bne.s  no_get              ;Someone else got it
;
; We just stole the serial port registers.  Wait forever.
; Nobody else can use the serial port, including the serial.device!
```

```
             move.l _AbsExecBase,a6
             move.l #0,d0                ;Wait for nothing (forever)
             JSRLIB Wait
no_get                                   ;Someone else has it, exit!
no_open      move.l #21,d0
             rts

MiscName     dc.b    'misc.resource',0
MyName       dc.b    'Serial Port hog',0
             END
```

```
/* An example of using the potgo.resource to read pins 9 and 5 of
* port 1 (the non-mouse port).  This bypasses the gameport.device.
* When the right button on a mouse plugged into port 1 is pressed,
* the read value will change. Use of port 0 (mouse) is unaffected.
*/
#include "exec/types.h"
#include "libraries/dos.h"

APTR  PotgoBase;
ULONG potbits;
UWORD value;

#define UNLESS(x) if(!(x))
#define UNTIL(x)  while(!(x))
#define OUTRY 1L<<15
#define DATRY 1L<<14
#define OUTRX 1L<<13
#define DATRX 1L<<12
                           /* Note: Aztec C prototypes use an extra version */
void main()                /* argument in the OpenResource() function       */
{
       UNLESS(PotgoBase=(APTR)OpenResource("potgo.resource"))
             exit(RETURN_FAIL);
       printf("PotgoBase is at $%lx\n",PotgoBase);

       potbits=AllocPotBits(OUTRY|DATRY|OUTRX|DATRX);
       /* Get the bits for the right and middle mouse buttons
          on the alternate mouse port. */

       if(potbits != (OUTRY|DATRY|OUTRX|DATRX))
             {
             printf("Pot bits are already allocated! %lx\n",potbits);
             FreePotBits(potbits);
             exit(RETURN_FAIL+1);
             }

       WritePotgo(0xFFFFFFFFL,potbits);
       /* Set all ones in the register (masked by potbits) */

       UNTIL(SIGBREAKF_CTRL_C & SetSignal(0L,0L))
             /* until CTRL-C is pressed */
             {
             value=*(UWORD *)0x00DFF016;
             /* Read word at $DFF016 */
             printf("POTINP = $%lx\n",value & potbits);
             /* Show what was read (restricted to our allocated bits) */
             }
       FreePotBits(potbits);
}
```

battclock.resource Page 1

TABLE OF CONTENTS

battclock.resource/ReadBattClock()
battclock.resource/ResetBattClock()
battclock.resource/WriteBattClock()

battclock.resource Page 2

```
battclock.resource/ReadBattClock()          battclock.resource/ReadBattClock()

   NAME
        ReadBattClock -- Read time from clock chip.  (V36)

   SYNOPSIS
        AmigaTime = ReadBattClock( )

        ULONG ReadBattClock( void );
        D0

   FUNCTION
        This routine reads the time from the clock chip and returns it
        as the number of seconds from 01-jan-1978.

   INPUTS

   RESULTS
        AmigaTime       The number of seconds from 01-Jan-1978 that
                            the clock chip thinks it is.

   NOTES
        If the clock chip returns an invalid date, the clock chip is
        reset and 0 is returned.

   SEE ALSO

   BUGS
```

```
battclock.resource/ResetBattClock()          battclock.resource/ResetBattClock()

   NAME
        ResetBattClock -- Reset the clock chip.  (V36)

   SYNOPSIS
        ResetBattClock( )

        void ResetBattClock( void );

   FUNCTION
        This routine does whatever is neeeded to put the clock chip
        into a working and usable state and also sets the date on the
        clock chip to 01-Jan-1978.

   INPUTS

   RESULTS

   NOTES

   SEE ALSO

   BUGS
```

```
battclock.resource/WriteBattClock()          battclock.resource/WriteBattClock()

   NAME
        WriteBattClock -- Set the time on the clock chip.  (V36)

   SYNOPSIS
        WriteBattClock( AmigaTime )
                        D0

        void WriteBattClock( ULONG );

   FUNCTION
        This routine writes the time given in AmigaTime to the clock
        chip.

   INPUTS
        AmigaTime       The number of seconds from 01-Jan-1978 to the
                            time that should be written to the clock
                            chip.

   RESULTS

   NOTES

   SEE ALSO

   BUGS
```

battmem.resource Page 1

```
TABLE OF CONTENTS
```

```
battmem.resource/ObtainBattSemaphore()
battmem.resource/ReadBattMem()
battmem.resource/ReleaseBattSemaphore()
battmem.resource/WriteBattMem()
```

battmem.resource Page 2

```
battmem.resource/ObtainBattSemaphore() battmem.resource/ObtainBattSemaphore()

   NAME
        ObtainBattSemaphore -- Obtain access to nonvolatile ram. (V36)

   SYNOPSIS
        ObtainBattSemaphore( )

        void ObtainBattSemaphore( void );

   FUNCTION
        Aquires exclusive access to the system nonvolatile ram.

   INPUTS

   RESULTS

   NOTES

   SEE ALSO

   BUGS
```

```
battmem.resource/ReadBattMem()                    battmem.resource/ReadBattMem()

   NAME
       ReadBattMem -- Read a bitstring from nonvolatile ram.  (V36)

   SYNOPSIS
       Error = ReadBattMem( Buffer, Offset, Len )
       D0                   A0      D0      D1

       ULONG ReadBattMem( APTR, ULONG, ULONG );

   FUNCTION
       Read a bitstring from nonvolatile ram.

   INPUTS
       Buffer          Where to put the bitstring.
       Offset          Bit offset of first bit to read.
       Len             Length of bitstring to read.

   RESULTS
       Error           Zero if no error.

   NOTES
       The battery-backed memory is checksummed. If a checksum error
       is detected, all bits in the battery-backed memory are
       silently set to zero.

       Bits in the battery-backed memory that do not exist are read
       as zero.

       Partial byte reads (less than 8 bits) result in the bits read
       being put in the low-order bits of the destination byte.

   SEE ALSO

   BUGS
```

```
battmem.resource/ReleaseBattSemaphore() battmem.resource/ReleaseBattSemaphore()

   NAME
       ReleaseBattSemaphore -- Allow nonvolatile ram to others.  (V36)

   SYNOPSIS
       ReleaseBattSemaphore( )

       void ReleaseBattSemaphore( void );

   FUNCTION
       Relinquish exclusive access to the system nonvolatile ram.

   INPUTS

   RESULTS

   NOTES

   SEE ALSO

   BUGS
```

```
battmem.resource/WriteBattMem()              battmem.resource/WriteBattMem()

   NAME
        WriteBattMem -- Write a bitstring to nonvolatile ram.  (V36)

   SYNOPSIS
        Error = WriteBattMem( Buffer, Offset, Len )
        D0                    A0      D0      D1

        ULONG WriteBattMem( APTR, ULONG, ULONG );

   FUNCTION
        Write a bitstring to the nonvolatile ram.

   INPUTS
        Buffer          Where to get the bitstring.
        Offset          Bit offset of first bit to write.
        Len             Length of bitstring to write.

   RESULTS
        Error           Zero if no error.

   NOTES
        The battery-backed memory is checksummed. If a checksum error
        is detected, all bits in the battery-backed memory are
        silently set to zero.

        Partial byte writes (less than 8 bits) result in the bits
        written being read from the low-order bits of the source byte.

   SEE ALSO

   BUGS
```

TABLE OF CONTENTS

cia.resource/AbleICR()
cia.resource/AddICRVector()
cia.resource/RemICRVector()
cia.resource/SetICR()

```
cia.resource/AbleICR()                                 cia.resource/AbleICR()

   NAME
       AbleICR -- Enable/disable ICR interrupts.

   SYNOPSIS
       oldMask = AbleICR( mask )
       D0                 D0

       WORD AbleICR( WORD );

   FUNCTION
       This function provides a means of enabling and disabling 8520
       CIA interrupt control registers. In addition it returns the
       previous enable mask.

   INPUTS
       mask            A bit mask indicating which interrupts to be
                           modified. If bit 7 is clear the mask
                           indicates interrupts to be disabled. If
                           bit 7 is set, the mask indicates
                           interrupts to be enabled. Bit positions
                           are identical to those in 8520 ICR.

   RESULTS
       oldMask         The previous enable mask before the requested
                           changes. To get the current mask without
                           making changes, call the function with a
                           null parameter.

   EXAMPLES
       Get the current mask:
           mask = AbleICR(0)
       Enable both timer interrupts:
           AbleICR(0x83)
       Disable serial port interrupt:
           AbleICR(0x08)

   EXCEPTIONS
       Enabling the mask for a pending interrupt will cause an
       immediate processor interrupt (that is if everything else is
       enabled). You may want to clear the pending interrupts with
       SetICR() prior to enabling them.

   NOTE
       The CIA resources are special in that there is more than one
       of them in the system. Because of this, the C language stubs
       in amiga.lib for the CIA resources require an extra parameter
       to specify which CIA resource to use. The synopsys for the
       amiga.lib stubs is as follows:

       oldMask = AbleICR( Resource, mask )
       D0                 A6        D0

       WORD AbleICR( struct Library *, WORD );

   SEE ALSO
       cia.resource/SetICR()
```

cia.resource Page 3

```
cia.resource/AddICRVector()                             cia.resource/AddICRVector()

   NAME
       AddICRVector -- attach an interrupt handler to a CIA bit.

   SYNOPSIS
       interrupt = AddICRVector( iCRBit, interrupt )
       D0                        D0      A1

       struct Interrupt *AddICRVector( WORD, struct Interrupt *);

   FUNCTION
       Assign interrupt processing code to a particular interrupt bit
       of the CIA ICR.  If the interrupt bit has already been
       assigned, this function will fail, and return a pointer to the
       owner interrupt.  If it succeeds, a null is returned.

       This function will also enable the CIA interrupt for the given
       ICR bit.

   INPUTS
       iCRBit          Bit number to set (0..4).
       interrupt       Pointer to interrupt structure.

   RESULT
       interrupt       Zero if successful, otherwise returns a
                           pointer to the current owner interrupt
                           structure.

   NOTE
       A processor interrupt may be generated immediatly if this call
       is successful.

       In general, it is probably best to only call this function
       while DISABLED so that the resource to which the interrupt
       handler is being attached may be set to a known state before
       the handler is called. You MUST NOT change the state of the
       resource before attaching your handler to it.

       The CIA resources are special in that there is more than one
       of them in the system. Because of this, the C language stubs
       in amiga.lib for the CIA resources require an extra parameter
       to specify which CIA resource to use. The synopsys for the
       amiga.lib stubs is as follows:

       interrupt = AddICRVector( Resource, iCRBit, interrupt )
       D0                        A6        D0      A1

       struct Interrupt *AddICRVector( struct Library *, WORD,
               struct Interrupt *);

       ***WARNING***

       Never assume that any of the CIA hardware is free for use.
       Always use the AddICRVector() function to obtain ownership
       of the CIA hardware registers your code will use.

       Note that there are two (2) interval timers per CIA.  If
       your application needs one of the interval timers, you
       can try to obtain any one of the four (4) until AddICRVector()
       succeeds.  If all four interval timers are in-use, your
       application should exit cleanly.

       If you just want ownership of a CIA hardware timer, or register,
       but do not want interrupts generated, use the AddICRVector()
       function to obtain ownership, and use the AbleICR() function
```

cia.resource Page 4

```
       to turn off (or on) interrupts as needed.

       Note that CIA-B generates level 6 interrupts (which can degrade
       system performance by blocking lower priority interrupts).  As
       usual, interrupt handling code should be optimized for speed.

       Always call RemICRVector() when your code exits to release
       ownership of any CIA hardware obtained with AddICRVector().

   SEE ALSO
       cia.resource/RemICRVector(), cia.resource/AbleICR()
```

```
cia.resource/RemICRVector()                         cia.resource/RemICRVector()

   NAME
       RemICRVector -- Detach an interrupt handler from a CIA bit.

   SYNOPSIS
       RemICRVector( iCRBit, interrupt )
                     D0      A1

       void RemICRVector( WORD, struct Interrupt *);

   FUNCTION
       Disconnect interrupt processing code for a particular
       interrupt bit of the CIA ICR.

       This function will also disable the CIA interrupt for the
       given ICR bit.

   INPUTS
       iCRBit          Bit number to set (0..4).
       interrupt       Pointer to interrupt structure.

   RESULT

   NOTE
       The CIA resources are special in that there is more than one
       of them in the system. Because of this, the C language stubs
       in amiga.lib for the CIA resources require an extra parameter
       to specify which CIA resource to use. The synopsys for the
       amiga.lib stubs is as follows:

       RemICRVector( Resource, iCRBit, interrupt )
                     A6        D0      A1

       void RemICRVector( struct Library *, WORD, struct Interrupt *);

   SEE ALSO
       cia.resource/AddICRVector()
```

```
cia.resource/SetICR()                                    cia.resource/SetICR()

   NAME
       SetICR -- Cause, clear, and sample ICR interrupts.

   SYNOPSIS
       oldMask = SetICR( mask )
       D0                D0

       WORD SetICR( WORD );

   FUNCTION
       This function provides a means of reseting, causing, and
       sampling 8520 CIA interrupt control registers.

   INPUTS
       mask            A bit mask indicating which interrupts to be
                           effected. If bit 7 is clear the mask
                           indicates interrupts to be reset.  If bit
                           7 is set, the mask indicates interrupts to
                           be caused. Bit positions are identical to
                           those in 8520 ICR.

   RESULTS
       oldMask         The previous interrupt register status before
                           making the requested changes.  To sample
                           current status without making changes,
                           call the function with a null parameter.

   EXAMPLES
       Get the interrupt mask:
           mask = SetICR(0)
       Clear serial port interrupt:
           SetICR(0x08)

   NOTE
       The CIA resources are special in that there is more than one
       of them in the system. Because of this, the C language stubs
       in amiga.lib for the CIA resources require an extra parameter
       to specify which CIA resource to use. The synopsys for the
       amiga.lib stubs is as follows:

       oldMask = SetICR( Resource, mask )
       D0                A6        D0

       WORD SetICR( struct Library *, WORD );

       ***WARNING***

       Never read the contents of the CIA interrupt control registers
       directly.  Reading the contents of one of the CIA interrupt
       control registers clears the register.  This can result in
       interrupts being missed by critical operating system code, and
       other applications.

   EXCEPTIONS
       Setting an interrupt bit for an enabled interrupt will cause
       an immediate interrupt.

   SEE ALSO
       cia.resource/AbleICR()
```

TABLE OF CONTENTS

disk.resource/AllocUnit
disk.resource/FreeUnit
disk.resource/GetUnit
disk.resource/GetUnitID
disk.resource/GiveUnit
disk.resource/ReadUnitID

```
disk.resource/AllocUnit                                   disk.resource/AllocUnit

   NAME
       AllocUnit - allocate a unit of the disk

   SYNOPSIS
       Success = AllocUnit( unitNum ), DRResource
        D0                   D0          A6

        BOOL AllocUnit(LONG);

   FUNCTION
        This routine allocates one of the units of the disk.  It should
        be called before trying to use the disk (via GetUnit).

        In reality, it is perfectly fine to use GetUnit/GiveUnit if AllocUnit
        fails.  Do NOT call FreeUnit if AllocUnit did not succeed.  This
        has been the case for all revisions of disk.resource.

   INPUTS
        unitNum -- a legal unit number (zero through three)

   RESULTS
        Success -- nonzero if successful.  zero on failure.

   EXCEPTIONS

   SEE ALSO

   BUGS
```

```
disk.resource/FreeUnit                                    disk.resource/FreeUnit

   NAME
       FreeUnit - deallocate the disk

   SYNOPSIS
       FreeUnit( unitNum ), DRResource
                  D0          A6

        void FreeUnit(LONG);

   FUNCTION
        This routine deallocates one of the units of the disk.  It should
        be called when done with the disk.  Do not call it if you did
        no successfully allocate the disk (there is no protection -- you
        will probably crash the disk system).

   INPUTS
        unitNum -- a legal unit number (zero through three)

   RESULTS

   EXCEPTIONS

   SEE ALSO
        FreeUnit

   BUGS
        Doesn't check if you own the unit, or even if anyone owns it.
```

```
disk.resource/GetUnit                                      disk.resource/GetUnit

   NAME
       GetUnit - allocate the disk for a driver

   SYNOPSIS
       lastDriver = GetUnit( unitPointer ), DRResource
       D0                    A1             A6

        struct DiscResourceUnit *GetUnit(struct DiscResourceUnit *);

   FUNCTION
        This routine allocates the disk to a driver.  It is either
        immediately available, or the request is saved until the disk
        is available.  When it is available, your unitPointer is
        sent back to you (via ReplyMsg).  You may then reattempt the
        GetUnit.

        Allocating the disk allows you to use the disk's resources.
        Remember however that there are four units to the disk; you are
        only one of them.  Please be polite to the other units (by never
        selecting them, and by not leaving interrupts enabled, etc.).

        When you are done, please leave the disk in the following state:
            dmacon dma bit ON
            dsklen dma bit OFF (write a #DSKDMAOFF to dsklen)
            adkcon disk bits -- any way you want
            entena:disk sync and disk block interrupts -- Both DISABLED
            CIA resource index interrupt -- DISABLED
            8520 outputs -- doesn't matter, because all bits will be
                set to inactive by the resource.
            8520 data direction regs -- restore to original state.

        NOTE: GetUnit() does NOT turn on the interrupts for you.
              You must use AbleICR (for the index interrupt) or intena
              (for the diskbyte and diskblock interrupts) to turn them
              on.  You should turn them off before calling GiveUnit,
              as stated above.

   INPUTS
        unitPtr - a pointer you your disk resource unit structure.
                Note that the message filed of the structure MUST
                be a valid message, ready to be replied to.  Make sure
                ln_Name points to a null-terminated string, preferably
                one that identifies your program.

                You need to set up the three interrupt structures,
                in particular the IS_DATA and IS_CODE fields.  Set them
                to NULL if you don't need that interrupt.  Also, set
                the ln_Type of the interrupt structure to NT_INTERRUPT.
                WARNING: don't turn on a disk resource interrupt unless
                the IS_CODE for that interrupt points to executable code!

                IS_CODE will be called with IS_DATA in A1 when the
                interrupt occurs.  Preserve all regs but D0/D1/A0/A1.
                Do not make assumptions about A0.

   RESULTS
        lastDriver - if the disk is not busy, then the last unit
                to use the disk is returned.  This may be used to
                see if a driver needs to reset device registers.
                (If you were the last user, then no one has changed
                any of the registers.  If someone else has used it,
                then any allowable changes may have been made).  If the
                disk is busy, then a null is returned.
```

```
EXCEPTIONS

SEE ALSO
     GiveUnit

BUGS
```

```
disk.resource/GetUnitID                                disk.resource/GetUnitID

   NAME
       GetUnitID - find out what type of disk is out there

   SYNOPSIS
       idtype = GetUnitID( unitNum ), DRResource
        D0                   D0         A6

        LONG GetUnitID(LONG);

   FUNCTION
        Gets the drive ID for a given unit.  Note that this value may
        change if someone calls ReadUnitID, and the drive id changes.

   INPUTS
        unitNum -- a legal unit number (zero through three)

   RESULTS
        idtype -- the type of the disk drive.  Standard types are
                defined in the resource include file.

   EXCEPTIONS

   SEE ALSO
        ReadUnitID

   BUGS
```

```
disk.resource/GiveUnit                             disk.resource/GiveUnit

   NAME
       GiveUnit - Free the disk back up

   SYNOPSIS
       GiveUnit(), DRResource
                   A6

        void GiveUnit();

   FUNCTION
        This routine frees the disk after a driver is done with it.
        If others are waiting, it will notify them.

   INPUTS

   RESULTS

   EXCEPTIONS

   SEE ALSO
        GetUnit

   BUGS
        In pre-V36, GiveUnit didn't check if you owned the unit.  A patch
        for this was part of 1.3.1 SetPatch.  Fixed in V36.
```

```
disk.resource/ReadUnitID                         disk.resource/ReadUnitID

   NAME
       ReadUnitID - reread and return the type of drive (V37)

   SYNOPSIS
       idtype = ReadUnitID( unitNum ), DRResource
        D0                  D0          A6

        ULONG ReadUnitID(LONG);

   FUNCTION
        Rereads the drive id for a specific unit (for handling drives
        that change ID according to what sort of disk is in them.  You
        MUST have done a GetUnit before calling this function!

   INPUTS
        unitNum -- a legal unit number (zero through three)

   RESULTS
        idtype -- the type of the disk drive.  Standard types are
                defined in the resource include file.

   EXCEPTIONS

   SEE ALSO
        GetUnitID

   BUGS
```

FileSystem.resource Page 1

TABLE OF CONTENTS

FileSystem.resource/--background--

FileSystem.resource Page 2

```
FileSystem.resource/--background--          FileSystem.resource/--background--

   PURPOSE

       The FileSystem.resource is where boot disk drivers rendezvous
       to share file system code segments for partitions specified by
       dos type.  Prior to V36, it was created by the first driver
       that needed to use it.  For V36, its creation is ensured by the
       rom boot process.

   CONTENTS

       The FileSystem.resource is described in the include file
       resources/filesysres.h.  The nodes on it describe how to
       algorithmically convert the result of MakeDosNode (from the
       expansion.library) to a node appropriate for the dos type.

       FileSysEntry
           fse_Node            on fsr_FileSysEntries list
                               ln_Name is of creator of this entry
           fse_DosType         DosType of this FileSys: e.g. 0x444f5301
                               for the fast file system.
           fse_Version         high word is the version, low word is
                               the revision.
           fse_PatchFlags      bits set for those of the following that
                               need to be substituted into a standard
                               device node for this file system: e.g.
                               $180 for substitute SegList & GlobalVec
           fse_Type            device node type: zero
           fse_Task            standard dos "task" field
           fse_Lock            must be zero
           fse_Handler         for V36, if bit 31 is set, this is not
                               an AmigaDOS partition.
           fse_StackSize       stacksize to use when starting task
           fse_Priority        task priority when starting task
           fse_Startup         startup msg: FileSysStartupMsg for disks
           fse_SegList         segment of code to run to start new task
           fse_GlobalVec       BCPL global vector when starting task

       no more entries need exist than those implied by fse_PatchFlags,
       so entries do not have a fixed size.  For V36, for example, the
       entry for the fast file system (fse_DosType 0x444f5301)
       contains a zero fse_PatchFlags, and thus no entries beyond that.
```

TABLE OF CONTENTS

misc.resource/AllocMiscResource
misc.resource/FreeMiscResource

```
misc.resource/AllocMiscResource                  misc.resource/AllocMiscResource

    NAME
        AllocMiscResource - allocate one of the miscelaneous resources

   SYNOPSIS
        CurrentUser = AllocMiscResource( unitNum, name )
        D0                               D0       A1

        char * AllocMiscResource(ULONG,char *);

   FUNCTION
        This routine attempts to allocate one of the miscellaneous resources
        If the resource had already been allocated, an error is returned. If
        you do get it, your name is associated with the resource (so a user
        can see who has it allocated).

        This function may not be called from interrupt code

   DESCRIPTION
        There are certain parts of the hardware that a multitasking- friendly
        program may need to take over.  The serial port is a good example. By
        grabbing the misc.resource for the serial port, the caller would
        "own" the hardware registers associated with that function.  Nobody
        else, including the system serial driver, is allowed to interfere.

        Resources are called in exactly the same manner as libraries.
        From assembly language, A6 must equal the resource base.  The
        offsets for the function are listed in the resources/misc.i
        include file (MR_ALLOCMISCRESOURCE for this function).

   INPUTS
        unitNum - the number of the resource you want to allocate
                   (eg.  MR_SERIALBITS).
        name - a mnenonic name that will help the user figure out
                what piece of software is hogging a resource.
                (havoc breaks out if a name of null is passed in...)

   RESULTS
        CurrentUser - if the resource is busy, then the name of
                the current user is returned.  If the resource is
                free, then null is returned.

   BUGS

   SEE ALSO
        resources/misc.i, FreeMiscResource()
```

```
misc.resource/FreeMiscResource                    misc.resource/FreeMiscResource

   NAME
        FreeMiscResource - make a resource available for reallocation

   SYNOPSIS
        FreeMiscResource( unitNum )
                          D0

        void FreeMiscResource(ULONG);

   FUNCTION
        This routine frees one of the resources allocated
        by AllocMiscResource.  The resource is made available
        for reuse.

        FreeMiscResource must be called from the same task that
        called AllocMiscResource.  This function may not be called from
        interrupt code.

    INPUTS
        unitNum - the number of the miscellaneous resource to be freed.

    RESULTS
        Frees the appropriate resource.

    BUGS

    SEE ALSO
        resources/misc.i, AllocMiscResource()
```

TABLE OF CONTENTS

potgo.resource/AllocPotBits
potgo.resource/FreePotBits
potgo.resource/WritePotgo

```
potgo.resource/AllocPotBits                         potgo.resource/AllocPotBits

   NAME
       AllocPotBits -- Allocate bits in the potgo register.

   SYNOPSIS
       allocated = AllocPotBits(bits)
       D0                       D0

       UWORD AllocPotBits( UWORD );

   FUNCTION
       The AllocPotBits routine allocates bits in the hardware potgo
       register that the application wishes to manipulate via
       WritePotgo.  The request may be for more than one bit.  A
       user trying to allocate bits may find that they are
       unavailable because they are already allocated, or because
       the start bit itself (bit 0) has been allocated, or if
       requesting the start bit, because input bits have been
       allocated.  A user can block itself from allocation: i.e.
       it should FreePotgoBits the bits it has and re-AllocPotBits if
       it is trying to change an allocation involving the start bit.

   INPUTS
       bits - a description of the hardware bits that the application
               wishes to manipulate, loosely based on the register
               description itself:
           START (bit 0) - set if you wish to use start (i.e. start
                   the proportional controller counters) with the
                   input ports you allocate (below).  You must
                   allocate all the DATxx ports you want to apply
                   START to in this same call, with the OUTxx bit
                   clear.
           DATLX (bit 8) - set if you wish to use the port associated
                   with the left (0) controller, pin 5.
           OUTLX (bit 9) - set if you promise to use the LX port in
                   output mode only.  The port is not set to output
                   for you at this time -- this bit set indicates
                   that you don't mind if STARTs are initiated at any
                   time by others, since ports that are enabled for
                   output are unaffected by START.
           DATLY (bit 10) - as DATLX but for the left (0) controller,
                   pin 9.
           OUTLY (bit 11) - as OUTLX but for LY.
           DATRX (bit 12) - the right (1) controller, pin 5.
           OUTRX (bit 13) - OUT for RX.
           DATRY (bit 14) - the right (1) controller, pin 9.
           OUTRY (bit 15) - OUT for RY.

   RESULTS
       allocated - the START and DATxx bits of those requested that
               were granted.  The OUTxx bits are don't cares.
```

potgo.resource

Page 3

```
potgo.resource/FreePotBits                             potgo.resource/FreePotBits

   NAME
        FreePotBits -- Free allocated bits in the potgo register.

   SYNOPSIS
        FreePotBits(allocated)
                    D0

        void FreePotBits( UWORD );

   FUNCTION
        The FreePotBits routine frees previously allocated bits in the
        hardware potgo register that the application had allocated via
        AllocPotBits and no longer wishes to use.  It accepts the
        return value from AllocPotBits as its argument.
```

---

potgo.resource

Page 4

```
potgo.resource/WritePotgo                               potgo.resource/WritePotgo

   NAME
        WritePotgo -- Write to the hardware potgo register.

   SYNOPSIS
        WritePotgo(word, mask)
                   D0    D1

        void WritePotgo( UWORD, UWORD );

   FUNCTION
        The WritePotgo routine sets and clears bits in the hardware
        potgo register.  Only those bits specified by the mask are
        affected -- it is improper to set bits in the mask that you
        have not successfully allocated.  The bits in the high byte
        are saved to be maintained when other users write to the
        potgo register.  The START bit is not saved, it is written
        only explicitly as the result of a call to this routine with
        the START bit set: other users will not restart it.

   INPUTS
        word - the data to write to the hardware potgo register and
            save for further use, except the START bit, which is
            not saved.
        mask - those bits in word that are to be written.  Other
            bits may have been provided by previous calls to
            this routine, and default to zero.
```

Linker library Autodocs

This section contains Autodoc summaries for the amiga.lib and debug.lib linker libraries, and reference source code listings for Exec support functions in amiga.lib. Unlike the libraries described in the first section of this book, the linker libraries are not shared run-time libraries. Instead, they are concatenated Amiga format object modules which are linked with your code as library files. The linker scans specified library files and inserts a copy of each referenced library function into your program code.

The linker libraries described here are debug.lib and amiga.lib:

- ☐ amiga.lib is the main Amiga scanned linker library, generally linked with every program for the Amiga. The major components of amiga.lib are:

  stubs - Individual interface stubs for each Amiga ROM routine that enable stack-based C compilers to call register-based Amiga ROM routines.

  offsets - The negative Library Vector Offset (_LVO) for each Amiga function. Assembly language program should link with amiga.lib to resolve their _LVO references.

  exec_support - C functions that simplify many Exec procedures such as the creation and deletion of tasks, ports, and I/O request structures. Source code is provided for these functions.

  clib - C support functions including pseudo-random number generation and a limited set of file and stdio functions designed to work directly with AmigaDOS file handles.

  other - Miscellaneous handy functions, callable from any language.

- debug.lib contains stdio-like functions for communicating with a serial terminal connected to the Amiga via its built-in serial port. Typically, this terminal will be a 9600 baud, 8 data bits, one stop bit connection to an external terminal or an Amiga running a terminal package. The debug.lib functions allow you to output messages and prompt for input, even from within low-level task or interrupt code, without disturbing the Amiga's display and or current state (other than the state of the serial hardware itself). No matter how badly the system may have crashed, these functions can usually get a message out.

  A similar debugging library currently called ddebug.lib is available for sending debugging output to the parallel port. This is useful for debugging serial applications. Ddebug.lib is not documented here. It contains functions similar to debug.lib but with names starting with 'd' instead of 'k'.

Shown below is a simple example of using the linker libraries from assembly language.

```
*  Demonstrates assembler use of the compiled C exec support routines
*  (CreatePort, etc.) in amiga.lib, and also the use of amiga.lib
*  csupport functions such as _printf for simple formatted output and
*  debugging.  Creates a port, outputs its address, and deletes the port.
*  Comments show the functions as they would be called from C.
*
*  LINK INSTRUCTIONS: Alink with Astartup.obj ... LIBRARY amiga.lib
*   Astartup sets up DOSBase and the stdout needed for amiga.lib _printf.
*   If you do not link with Astartup.obj, you must add the following
*   variables, XDEF them, and initialize them as commented:
*   DC.L  _DOSBase   0  ;needs base returned from OpenLibrary of dos.library
*   DC.L  _stdout    0  ;needs an AmigaDOS file handle from a dos Open call
*   DC.L  _SysBase   0  ;needs the address stored at location 4

       INCLUDE "exec/types.i"
       INCLUDE "exec/io.i"
       INCLUDE "libraries/dos.i"

*------ Imported labels:  C interface Amiga.lib routines
       XREF  _CreatePort
       XREF  _DeletePort
       XREF  _printf

*------ Exported labels:  Where Astartup.obj JSR's to our code
       XDEF   _main

             CODE

;use startup code (_main + link with Astartup.obj)
_main:
             movem.l   d2-d7/a2-a6,-(sp)  ;Save registers
```

```
*----- Exec Support function:  msgPort = CreatePort(name,pri)

             move.l    #0,-(sp)      ;push priority 0 on stack as long
             pea       portname      ;push addr of null-termed portname
             jsr       _CreatePort   ;call CreatePort
             addq.l    #8,sp         ;add 4 to stack for each long pushed
             jsr       mydebug0      ;rtn to print d0 (preserves d0)
             tst.l     d0            ;test result
             beq.s     failure       ;if zero, CreatePort failed

*----- Exec Support function:  DeletePort(port)
             move.l    d0,-(sp)         ;else push d0 (now our msgPort)
             jsr       _DeletePort      ;call DeletePort
             addq.l    #4,sp            ;add 4 to stack for pushed long

             move.l    #RETURN_OK,d0    ;set up success return code
             bra.s     endcode          ;and skip to exit code

*----- Failure to CreatePort branches here
failure:
             move.l    #RETURN_FAIL,d0      ;set up failure return code


endcode:
             movem.l   (sp)+,d2-d7/a2-a6    ;Restore registers
             rts                            ;rts with d0 = return code

*----- mydebug0 - Subroutine uses Amiga.lib _printf to print the contents
*                 of d0.  Preserves all registers.
mydebug0:
             movem.l   d0-d7/a0-a6,-(sp)    ;save registers

*----- C Support function printf(): here  printf("$%lx\n",contents_of_d0)
*      Note that the fstrl DC.B below specifies '\n' and null as 10,0

             move.l    d0,-(sp)              ;push d0 on the stack
             pea       fstrl                 ;push addr of format string
             jsr       _printf               ;call printf
             addq.l    #8,sp                 ;add 4 to stack for each long
             movem.l   (sp)+,d0-d7/a0-a6     ;restore saved registers
             rts                             ;rts

             DATA
portname     DC.B  'sample_msgport',0
fstrl        DC.B  '$%lx',10,0
             END


;--------------------------------------------------------------------
; Example C Callable function that adds two numbers.  From C, the
; call would look like this:
;            result=AddThemUp(first,second);
;
             XDEF   _AddThemUp   ;Make an External Definition
_AddThemUp
             move.l 4(sp),D0     ;Get FIRST number
             move.l 8(sp),D1     ;Get SECOND number
             add.l  D1,D0        ;Add them
             rts                 ;Return result (which is in D0)
```

TABLE OF CONTENTS

amiga.lib/AddTOF
amiga.lib/BeginIO
amiga.lib/CreateExtIO
amiga.lib/CreatePort
amiga.lib/CreateTask
amiga.lib/DeleteExtIO
amiga.lib/DeletePort
amiga.lib/DeleteTask
amiga.lib/FastRand
amiga.lib/math/afp
amiga.lib/math/arnd
amiga.lib/math/dbf
amiga.lib/math/fpa
amiga.lib/math/fpbcd
amiga.lib/NewList
amiga.lib/printf
amiga.lib/RangeRand
amiga.lib/RemTOF
amiga.lib/sprintf
amiga.lib/stdio

```
amiga.lib/AddTOF                                          amiga.lib/AddTOF

   NAME
       AddTOF - add a task to the TopOfFrame Interrupt server chain.

   SYNOPSIS
       AddTOF(i,p,a);
        void AddTOF(struct Isrvstr *, APTR, APTR);

   FUNCTION
       Adds a task to the vertical-blanking interval interrupt server
       chain.  This prevents C programmers from needing to write an
       assembly language stub to do this function.

   INPUTS
       i - pointer to structure Isrvstr.
       p - pointer to the C-code routine that this server is to call each
           time TOF happens.
       a - pointer to the first longword in an array of longwords that
           is to be used as the arguments passed to your routine
           pointed to by p.

   SEE ALSO
       RemTOF, graphics/graphint.h
```

amiga.lib.library Page 3

```
amiga.lib/BeginIO                                               amiga.lib/BeginIO

   NAME
       BeginIO -- initiate asynchronous I/O

   SYNOPSIS
       BeginIO(iORequest)
       void BeginIO(struct IORequest *);

   FUNCTION
       This function takes an IORequest, and passes it directly to the
       BEGINIO vector of the proper device.  This works exactly like
       SendIO, but does not clear the io_Flags field first.

       This function does not wait for the I/O to complete.

   INPUTS
       iORequest - Pointer to an initialized, open IORequest structure
                   with the io_Flags field set to a reasonable value
                   (use zero if you do not require io_Flags).

   SEE ALSO
       exec/DoIO, exec/SendIO, exec/WaitIO
```

amiga.lib.library Page 4

```
amiga.lib/CreateExtIO                                       amiga.lib/CreateExtIO

   NAME
       CreateExtIO() -- create an IORequest structure

   SYNOPSIS
       ioReq = CreateExtIO( ioReplyPort, size );
       struct IORequest *CreateExtIO(struct MsgPort *, ULONG);

   FUNCTION
       Allocates memory for and initializes a new IO request block
       of a user-specified number of bytes.  The number of bytes
       MUST be the size of a legal IORequest (or extended IORequest)
       or very nasty things will happen.

   NOTE
       V36 Exec supports this feature with the CreateIORequest() function.

   INPUTS
       ioReplyPort - a pointer to an already initialized
               message port to be used for this IO request's reply port.
               (usually created by CreatePort()).
       size - the size of the IO request to be created.

   RESULT
       Returns a pointer to the new IO Request block, or NULL if
               the request failed.

   SEE ALSO
       CreatePort, DeleteExtIO
       CreateIORequest
```

amiga.lib.library Page 5

```
amiga.lib/CreatePort                                      amiga.lib/CreatePort

   NAME
       CreatePort - Allocate and initialize a new message port

   SYNOPSIS
       CreatePort(name,pri)
       struct MsgPort *CreatePort(char *,LONG);

   FUNCTION
       Allocates and initializes a new message port.  The message list
       of the new port will be prepared for use (via NewList).  The port
       will be set to signal your task when a message arrives (PA_SIGNAL).

   NOTE
       V36 Exec supports this feature with the CreateMsgPort() function.

   INPUTS
       name - NULL if other tasks will not search for this port
              via the FindPort() call.  If non-null, this must be
              a null-terminated string; the port will be added to
              the system public port list.  The name is not copied.
       pri  - Priority used for insertion into the public port list.

   RESULT
       A new MsgPort structure ready for use.

   SEE ALSO
       DeletePort, exec/FindPort, exec/ports.h
```

amiga.lib.library Page 6

```
amiga.lib/CreateTask                                        amiga.lib/CreateTask

   NAME
       CreateTask -- Create task with given name, priority, stacksize

   SYNOPSIS
       CreateTask( name, pri, initPC, stackSize)
       task=(struct Task *)CreateTask(char *, LONG, funcEntry, ULONG);

   FUNCTION
       This function simplifies program creation of subtasks by
       dynamically allocating and initializing required structures
       and stack space, and adding the task to Exec's task list
       with the given name and priority.  A tc_MemEntry list is provided
       so that all stack and structure memory allocated by CreateTask
       is automatically deallocated when the task is removed.

       An Exec task may not call dos.library functions or any function
       which might cause the loading of a disk-resident library, device,
       or file (since such functions are indirectly calls to dos.library).
       Only AmigaDOS Processes may call AmigaDOS; see the DOS CreateProc()
       call for more information.

       If other tasks or processes will need to find this task by name,
       provide a complex and unique name to avoid conflicts.

       If your compiler provides automatic insertion of stack-checking
       code, you may need to disable this feature when compiling subtask
       code since the stack for the subtask is at a dynamically allocated
       location.  If your compiler requires 68000 registers to contain
       particular values for base relative addressing, you may need to
       save these registers from your main process, and restore them
       in your initial subtask code.

       The function entry initPC is generally provided as follows:

       In C:
       extern void functionName();
       char *tname = "unique name";
       task = CreateTask(tname, 0L, functionName, 4000L);

       In assembler:
               PEA     startLabel

   INPUTS

       name - a null terminated string.
       pri - an Exec task priority between -128 and 127 (commonly 0)
       funcEntry - the address of the first executable instruction
                   of the subtask code.
       stackSize - size in bytes of stack for the subtask.  Don't cut it
                   too close - system function stack usage may change.

   SEE ALSO
       DeleteTask, exec/FindTask
```

```
amiga.lib/DeleteExtIO                                   amiga.lib/DeleteExtIO

   NAME
       DeleteExtIO() - return memory allocated for extended IO request

   SYNOPSIS
       DeleteExtIO( ioReq );
       void DeleteExtIO(struct IORequest *);

   NOTE
       V36 Exec supports this feature with the DeleteIORequest() function.

   FUNCTION
       Frees up an IO request as allocated by CreateExtIO().  By
       looking at the mn_Length field, it knows how much memory
       to deallocate.

   INPUTS
       ioReq - A pointer to the IORequest block to be freed.

   SEE ALSO
       CreateExtIO
```

```
amiga.lib/DeletePort                                     amiga.lib/DeletePort

   NAME
       DeletePort - Free a message port created by CreatePort

   SYNOPSIS
       DeletePort(msgPort)
       void DeletePort(struct MsgPort *);

   NOTE
       V36 Exec supports this feature with the DeleteMsgPort() function.

   FUNCTION
       Frees a message port created by CreatePort.  All messages that
       may have been attached to this port must have already been
       replied to.

   INPUTS
       msgPort - A message port

   SEE ALSO
       CreatePort
```

amiga.lib.library Page 9

```
amiga.lib/DeleteTask                                  amiga.lib/DeleteTask

   NAME
       DeleteTask -- Delete a task created with CreateTask

   SYNOPSIS
       DeleteTask( task )
       void DeleteTask(struct Task *);

   FUNCTION
       This function simply calls exec/RemTask, deleting a task from the
       Exec task lists and automatically freeing any stack and
       structure memory allocated for it by CreateTask.

       Before deleting a task, you must first make sure that the task is
       not currently executing any system code which might try to signal
       the task after it is gone.

       This can be accomplished by stopping all sources that might reference
       the doomed task, then causing the subtask execute a Wait(0L).  Another
       option is to have have the task DeleteTask()/RemTask() itself.

   INPUTS
       task - pointer to a Task

   SEE ALSO
       CreateTask, exec/RemTask
```

amiga.lib.library Page 10

```
amiga.lib/FastRand                                          amiga.lib/FastRand

   NAME
       FastRand - quickly generate a somewhat random integer

   SYNOPSIS
       number = FastRand(seed);
        ULONG FastRand(ULONG);

   FUNCTION
       C-implementation only.   Seed value is taken from stack, shifted
       left one position, exclusive-or'ed with hex value $1D872B41 and
       returned (D0).

   INPUTS
       seed - a 32-bit integer

   RESULT
       number - new random seed, a 32-bit value

   SEE ALSO
       RangeRand
```

```
amiga.lib/math/afp                                          amiga.lib/math/afp

   NAME
       afp - Convert ASCII string variable into fast floating point

   USAGE
      ffp_value = afp(string);

    FUNCTION
       Accepts the address of the ASCII string in C format that is
       converted into an FFP floating point number.

       The string is expected in this Format:
       {S}{digits}{'.'}{digits}{'E'}{S}{digits}
       <*******MANTISSA*******><***EXPONENT***>


       Syntax rules:
       Both signs are optional and are '+' or '-'. The mantissa must be
       present. The exponent need not be present. The mantissa may lead
       with a decimal point. The mantissa need not have a decimal point.
       Examples: All of these values represent the number fourty-two.
                    42           .042e3
                    42.         +.042e+03
                   +42.          0.000042e6
               0000042.00       420000e-4
                               420000.00e-0004

       Floating point range:
       Fast floating point supports the value zero and non-zero values
       within the following bounds -
                       18                      20
        9.22337177 x 10   > +number >  5.42101070 x 10
                       18                         -20
       -9.22337177 x 10   > -number > -2.71050535 x 10

       Precision:
       This conversion results in a 24 bit precision with guaranteed
       error less than or equal to one-half least significant bit.

       INPUTS
       string - Pointer to the ASCII string to be converted.


       OUTPUTS
       string - points to the character which terminated the scan
       equ - fast floating point equivalent
```

```
amiga.lib/math/arnd                                        amiga.lib/math/arnd

   NAME
        arnd - ASCII round of the provided floating point string

   USAGE
        arnd(place, exp, &string[0]);

   FUNCTION
        Accepts an ASCII string representing an FFP floating point
        number, the binary representation of the exponent of said
        floating point number and the number of places to round to.
        A rounding process is initiated, either to the left or right
        of the decimal place and the result placed back at the
        input address defined by &string[0].

   INPUTS
        place - integer representing number of decimal places to round to
        exp - integer representing exponent value of the ASCII string
        &string[0] - address where rounded ASCII string is to be placed
                     (16 bytes)

   RESULT
        &string[0] - rounded ASCII string

   BUGS
        None
```

amiga.lib.library Page 13

```
amiga.lib/math/dbf                                   amiga.lib/math/dbf

   NAME
        dbf - convert FFP dual-binary number to FFP format

   USAGE
        fnum = dbf(exp, mant);

   FUNCTION
        Accepts a dual-binary format (described below) floating point
        number and converts it to an FFP format floating point number.
        The dual-binary format is defined as:

                exp bit  16     = sign (0=>positive, 1=>negative)
                exp bits 15-0   = binary integer representing the base
                                          ten (10) exponent
                man             = binary integer mantissa

   INPUTS
        exp - binary integer representing sign and exponent
        mant - binary integer representing the mantissa

   RESULT
        fnum - converted FFP floating point format number

   BUGS
        None
```

amiga.lib.library Page 14

```
amiga.lib/math/fpa                                                 amiga.lib/math/fpa

   NAME
        fpa - convert fast floating point into ASCII string equivalent

   USAGE
        exp = fpa(fnum, &string[0]);

   FUNCTION
        Accepts an FFP number and the address of the ASCII string where it's
        onverted output is to be stored.  The number is converted to a NULL
        terminated ASCII string in and stored at the address provided.
        Additionally, the base ten (10) exporent in binary form is returned.

   INPUTS
        fnum       - Motorola Fast Floating Point number
        &string[0] - address for output of converted ASCII character string
                       (16 bytes)

   RESULT
        &string[0] - converted ASCII character string
        exp        - integer exponent value in binary form

   BUGS
        None
```

```
amiga.lib/math/fpbcd                                    amiga.lib/math/fpbcd

   NAME
        fpbcd - convert FFP floating point number to BCD format

   USAGE
        fpbcd(fnum, &string[0]);

   FUNCTION
        Accepts a floating point number and the address where the
        converted BCD data is to be stored.  The FFP number is
        converted and stored at the specified address in an ASCII
        form in accordance with the following format:

                MMMM S E S B

        Where:  M = Four bytes of BCD, each with two (2) digits of
                    the mantissa (8 digits)
                S = Sign of mantissa (0x00 = positive, 0xFF = negative)
                E = BCD byte for two (2) digit exponent
                S = Sign of exponent (0x00 = positive, 0xFF = negative)
                B = One (1) byte binary two's compliment representation
                        of the exponent

   INPUTS
        fnum - floating point number
        &string[0] - address where converted BCD data is to be placed

   RESULT
        &string[0] - converted BCD data
```

```
amiga.lib/NewList                                          amiga.lib/NewList

   NAME
        NewList -- prepare a list structure for use

   SYNOPSIS
        NewList(list*)
        void NewList(struct List *);

   FUNCTION
        Prepare a List structure for use; the list will be empty and
        ready to use.

        This function prepares the lh_Head, lh_Tail and lh_TailPred fields.
        You are responsible for initializing lh_Type.  Assembly programmers
        will want to use the NEWLIST macro instead.

   INPUTS
        list - Pointer to a List

   SEE ALSO
        exec/lists.h
```

```
amiga.lib/printf                                          amiga.lib/printf

   NAME
        printf - print a formatted output line to the standard output.

   SYNOPSIS
        printf( formatstring [,value [,values] ] );

   FUNCTION
        Format the output in accordance with specifications in the format
        string:

   INPUTS
        formatstring - a pointer to a null-terminated string describing the
                     output data, and locations for parameter substitutions.
        value(s) - numeric variables or addresses of null-terminated strings
                 to be added to the format information.

        The function printf can handle the following format conversions, in
        common with the normal C language call to printf:

        %c  - the next long word in the array is to be formatted
              as a character (8-bit) value
        %d  - the next long word in the array is to be formatted
              as a decimal number
        %x  - the next long word in the array is to be formatted
              as a hexadecimal number
        %s  - the next long word is the starting address of a
              null-terminated string of characters

    And "l" (small-L) character must be added between the % and the letter
    if the value is a long (32 bits) or if the compiler in use forces
    passed paramters to 32 bits.

    Floating point output is not supported.

    Following the %, you may also specify:

    o       an optional minus (-) sign that tells the formatter
            to left-justify the formatted item within the field
            width

    o       an optional field-width specifier... that is, how
            many spaces to allot for the full width of this
            item.  If the field width specifier begins with
            a zero (0), it means that leading spaces, ahead of
            the formatted item (usually a number) are to be
            zero-filled instead of blank-filled

    o       an optional period (.) that separates the width
            specifier from a maximum number of characters
            specifier

    o       an optional digit string (for %ls specifications
            only) that specifies the maximum number of characters
            to print from a string.

    See other books on C language programming for examples of the use
    of these formatting options (see "printf" in other books).

   NOTE
        The global "_stdout" must be defined, and contain a pointer to
        a legal AmigaDOS file handle.  Using the standard Amiga startup
        module sets this up.  In other cases you will need to define
        stdout, and assign it to some reasonable value (like what the
```

```
        AmigaDOS Output() call returns).  This code would set it up:

        ULONG stdout;
                stdout=Output();
```

```
amiga.lib/RangeRand                                        amiga.lib/RangeRand

   NAME
       RangeRand - To obtain a random number within a specific integer range
                   of 0 to value.

   SYNOPSIS
       number = RangeRand(value);

   FUNCTION
       RangeRand accepts a value from 1 to 65535, and returns a value
       within that range. (16-bit integer).  Note:  C-language implementation.

       Value is passed on stack as a 32-bit integer but used as though
       it is only a 16-bit integer.  Variable named RangeSeed is available
       beginning with V1.2 that contains the global seed value passed from
       call to call and thus can be changed by a program by declaring::

               extern ULONG RangeSeed;

   INPUTS
       value - integer in the range of 1 to 65535.

   RESULT
       number - pseudo random integer in the range of 1 to <value>.

   SEE ALSO
       FastRand
```

```
amiga.lib/RemTOF                                              amiga.lib/RemTOF

   NAME
       RemTOF - Remove a task from the TopOfFrame interrupt server chain.

   SYNOPSIS
       RemTOF(i);
       void RemTOF(struct Isrvstr *);

   FUNCTION
       To remove a task from the vertical-blanking interval interrupt server
       chain.

   INPUTS
       i - pointer to structure Isrvstr.

   SEE ALSO
       AddTOF,graphics/graphinit.h
```

amiga.lib.library Page 21

```
amiga.lib/sprintf                                      amiga.lib/sprintf

   NAME
       sprintf - format a C-like string into a string buffer

   SYNOPSIS
       sprintf( destination, formatstring [,value [, values] ] );

   FUNCTION
       perform string formatting identical to printf, but direct the output
       into a specific destination in memory.  This uses the ROM version
        of printf, so it is very small.

        Assembly programmers can call this by placing values on the
        stack, followed by a pointer to the formatstring, followed
        by a pointer to the destination string.

   INPUTS
       destination - the address of an area in memory into which the
                     formatted output is to be placed.
       formatstring - pointer to a null terminated string describing the
                      desired output formatting.
       value(s) - numeric information to be formatted into the output
                  stream.

   SEE ALSO
       printf, exec/RawDoFmt
```

amiga.lib.library Page 22

```
amiga.lib/stdio                                              amiga.lib/stdio

   NAMES
        fclose  - close file
        fgetc   - get a character from a file
        fprintf - format data to file (see exec.library/RawDoFmt)
        fputc   - put character to file
        fputs   - write string to file
        getchar - get a character from stdin
        printf  - put format data to stdout (see exec.library/RawDoFmt)
        putchar - put character to stdout
        puts    - put string to stdout, followed by newline
        sprintf - format data into string (see exec.library/RawDoFmt)

   FUNCTION
        These functions work much like the standard C functions of the same
        names.  The file I/O functions all use non-buffered AmigaDOS
        filehandles, and must not be mixed with the file I/O of any C
        compiler.  The names of these function match those found in many
        standard C libraries, when a name conflict occurs, the function is
        generally taken from the FIRST library that was specified on the
        linker's command line.  Thus to use these functions, specify
        the amiga.lib library first.

        To get a suitable AmigaDOS filehandle, the AmigaDOS Open()
        function must be used.

        All of the functions that write to stdout expect an appropriate
        filehandle to have been set up ahead of time.  Depending on
        your C compiler and options, this may have been done by the
        startup code.  Or it can be done manually:

   FROM C:
        extern ULONG stdout;
        /* Remove the extern if startup code did not define stdout */
                stdout=Output();

   FROM ASSEMBLY:
                XDEF    _stdout
                DC.L    _stdout ;<- Place result of dos.library Output() here.
```

TABLE OF CONTENTS

debug.lib/KCmpStr
debug.lib/KGetChar
debug.lib/KGetNum
debug.lib/KMayGetChar
debug.lib/KPrintF
debug.lib/KPutChar
debug.lib/KPutStr

debug.lib/KCmpStr debug.lib/KCmpStr

NAME

KCmpStr - compare two null terminated strings

SYNOPSIS

```
mismatch = KCmpStr(string1, string2)
D0         A0       A1
```

FUNCTION

string1 is compared to string2 using the ASCII coalating sequence. 0 indicates the strings are identical.

debug.library Page 3

```
debug.lib/KGetChar                                          debug.lib/KGetChar

   NAME
        KGetChar - get a character from the console
                   (defaults to the serial port at 9600 baud)

   SYNOPSIS
        char = KGetChar()
        D0

   FUNCTION
        busy wait until a character arrives from the console.
        KGetChar is the assembly interface, _KGetChar and _kgetc
        are the C interfaces.
```

debug.library Page 4

```
debug.lib/KGetNum                                            debug.lib/KGetNum

   NAME
        KGetNum - get a number from the console

   SYNOPSIS
        number = KGetNum()
        D0

   FUNCTION
        get a signed decimal integer from the console.  This will busy
        wait until the number arrives.
```

```
debug.lib/KMayGetChar                                  debug.lib/KMayGetChar

   NAME
        KMayGetChar - return a character if present, but don't wait
                      (defaults to the serial port at 9600 baud)

   SYNOPSIS
        flagChar = KMayGetChar()
        D0

   FUNCTION
        return either a -1, saying that there is no character present, or
        whatever character was waiting.  KMayGetChar is the assembly
        interface,  _KMayGetChar is the C interface.
```

```
debug.lib/KPrintF                                          debug.lib/KPrintF

   NAME
        KPrintF - print formatted data to the console
                  (defaults to the serial port at 9600 baud)

   SYNOPSIS
        KPrintF("format string",values)
                A0              A1

   FUNCTION
        print a formatted C-type string to the console.  See the
        exec RawDoFmt() call for the supported % formatting commands.

   INPUTS
        "format string" - A C style string with % commands to indicate
                          where paramters are to be inserted.
        values - A pointer to an array of paramters, to be inserted into
                 specified places in the string.

        KPutFmt and KPrintF are identical assembly interfaces that want the
        two pointers in registers.  _KPrintF and _kprintf are C interfaces
        that expect the format string pointer on the stack, and the
        paramters on the stack above that.

   SEE ALSO
        exec.library/RawDoFmt, any C compiler's "printf" call.
```

debug.library Page 7

```
debug.lib/KPutChar                                         debug.lib/KPutChar

   NAME
        KPutChar - put a character to the console
                   (defaults to the serial port at 9600 baud)

   SYNOPSIS
        char = KPutChar(char)
        D0                D0

   FUNCTION
        Put a character to the console.  This function will not return
        until the character has been completely transmitted.

   INPUTS
        KPutChar is the assembly interface, the character must be in D0.
        _KPutChar and _kputc are the C interfaces, the character must be
        a longword on the stack.
```

debug.library Page 8

```
debug.lib/KPutStr                                           debug.lib/KPutStr

   NAME
        KPutStr - put a string to the console
                  (defaults to the serial port at 9600 baud)

   SYNOPSIS
        KPutStr(string)
                A0

   FUNCTION
        put a null terminated string to the console.  This function will
        not return until the string has been completely transmitted.

   INPUTS
        KPutStr is the assembly interface, a string pointer must be in A0.
        _KPutStr and _kputs are the C interfaces, the string pointer must
        be on the stack.
```

amiga.lib support functions Page 1

```
/****** amiga.lib/CreateExtIO *************************************/

#include "exec/types.h"
#include "exec/memory.h"
#include "exec/io.h"
/*
#include "proto/exec.h"
#include "functions.h"
*/


struct IORequest *CreateExtIO( ioReplyPort, size )
struct MsgPort *ioReplyPort;
ULONG  size;
{
struct IORequest *ioReq;

    if(! ioReplyPort )
        return(NULL);

    ioReq =
         (struct IORequest *)AllocMem( size, (ULONG)MEMF_CLEAR|MEMF_PUBLIC );

    if(!ioReq)
        return(NULL);

    ioReq->io_Message.mn_Node.ln_Type = NT_MESSAGE;
    ioReq->io_Message.mn_Length       = size;   /* save for later */
    ioReq->io_Message.mn_ReplyPort    = ioReplyPort;

    return( ioReq );
}



/****** amiga.lib/DeleteExtIO *************************************/

void DeleteExtIO( ioExt )
struct IORequest *ioExt;
{
    /* try to make it hard to reuse the request by accident */
    ioExt->io_Message.mn_Node.ln_Type = -1;
    ioExt->io_Message.mn_ReplyPort    = (struct MsgPort *)-1;
    ioExt->io_Device                  = (struct Device *) -1;

    FreeMem( ioExt, (ULONG)ioExt->io_Message.mn_Length );
}
```

amiga.lib support functions Page 2

```
/****** amiga.lib/CreatePort *************************************/

#include "exec/types.h"
#include "exec/ports.h"
#include "exec/memory.h"
/* #include "proto/exec.h"
   #include "functions.h"
*/

/* Example only, always use the amiga.lib version */

struct MsgPort *CreatePort(name, pri)
char *name;
LONG  pri;
{
int sigBit;
struct MsgPort *port;

    if ((sigBit = AllocSignal(-1L)) == -1)
        return(NULL);

    port = (struct MsgPort *)
      AllocMem((ULONG)sizeof(struct MsgPort),(ULONG)MEMF_CLEAR|MEMF_PUBLIC);

    if (!port)
        {
        FreeSignal(sigBit);
        return(NULL);
        }

    port-> mp_Node.ln_Name = name;
    port-> mp_Node.ln_Pri  = pri;
    port-> mp_Node.ln_Type = NT_MSGPORT;

    port-> mp_Flags   = PA_SIGNAL;
    port-> mp_SigBit  = sigBit;
    port-> mp_SigTask = (struct Task *)FindTask(0L); /* find THIS task */

    if (name)
        AddPort(port);
    else
        NewList(&(port-> mp_MsgList)); /* init message list */

    return(port);
}


/****** amiga.lib/DeletePort *************************************/

void DeletePort(port)
struct MsgPort *port;
{
    if ( port-> mp_Node.ln_Name ) /* if it was public... */
        RemPort(port);

    /* Make it difficult to re-use the port */
    port-> mp_SigTask        = (struct Task *) -1;
    port-> mp_MsgList.lh_Head = (struct Node *) -1;

    FreeSignal( port-> mp_SigBit );

    FreeMem( port, (ULONG)sizeof(struct MsgPort) );
}
```

amiga.lib support functions Page 3

```
/****** amiga.lib/CreateTask ***************************/

#include "exec/types.h"
#include "exec/tasks.h"
#include "exec/memory.h"

#include "proto/exec.h"
/*
#include "functions.h"
*/

/* the template for the mementries.  Unfortunately, this is hard to
 * do from C; mementries have unions, and they cannot be statically
 * initialized...
 *
 * In the interest of simplicity I recreate the mem entry structures
 * here with appropriate sizes.  We will copy this to a local
 * variable and set the stack size to what the user specified,
 * then attempt to actually allocate the memory.
 */
#define ME_TASK         0
#define ME_STACK        1
#define NUMENTRIES      2

struct FakeMemEntry {
    ULONG fme_Reqs;
    ULONG fme_Length;
};

struct FakeMemList {
    struct Node fml_Node;
    UWORD       fml_NumEntries;
    struct FakeMemEntry fml_ME[NUMENTRIES];
} TaskMemTemplate = {
    { 0 },                                        /* Node */
    NUMENTRIES,                                   /* num entries */
    {                                             /* actual entries: */
        { MEMF_PUBLIC | MEMF_CLEAR, sizeof( struct Task ) },    /* task */
        { MEMF_CLEAR,   0 }                                     /* stack */
    }
};



struct Task * CreateTask( name, pri, initPC, stackSize )
char  *name;
ULONG pri;
APTR  initPC;
ULONG stackSize;
{
struct Task *newTask;
struct FakeMemList fakememlist;
struct MemList *ml;

    /* round the stack up to longwords... */
    stackSize = (stackSize +3) & ~3;

    /*
     * This will allocate two chunks of memory: task of PUBLIC
     * and stack of PRIVATE
     */
    fakememlist = TaskMemTemplate;
    fakememlist.fml_ME[ME_STACK].fme_Length = stackSize;

    ml = (struct MemList *)AllocEntry( (struct MemList *)&fakememlist );
```

amiga.lib support functions Page 4

```
    if(! ml )
        return( NULL );

    /* set the stack accounting stuff */
    newTask = (struct Task *) ml->ml_ME[ME_TASK].me_Addr;

    newTask->tc_SPLower = ml->ml_ME[ME_STACK].me_Addr;
    newTask->tc_SPUpper = (APTR)((ULONG)(newTask->tc_SPLower) + stackSize);
    newTask->tc_SPReg   = newTask->tc_SPUpper;

    /* misc task data structures */
    newTask->tc_Node.ln_Type = NT_TASK;
    newTask->tc_Node.ln_Pri  = pri;
    newTask->tc_Node.ln_Name = name;

    /* add it to the tasks memory list */
    NewList( &newTask->tc_MemEntry );
    AddHead( &newTask->tc_MemEntry, (struct Node *)ml );

    /* add the task to the system -- use the default final PC */
    AddTask( newTask, initPC, 0L );
    return( newTask );
}


/****** amiga.lib/DeleteTask **********************************/

void DeleteTask( tc )
struct Task *tc;
{
    /* because we added a MemList structure to the tasks's TC_MEMENTRY
     * structure, all the memory will be freed up for us! */
    RemTask( tc );
}
```

amiga.lib support functions Page 5

```
****** amiga.lib/BeginIO ************************

        INCLUDE "exec/types.i"
        INCLUDE "exec/lists.i"
        INCLUDE "exec/io.i"

;Call the BeginIO vector of a device directly.  Much like exec/SendIO, but
;does not touch IO_FLAGS.
                SECTION _BeginIO
                XDEF    _BeginIO

_BeginIO:       move.l  4(sp),a1         ;Get IORequest pointer
                move.l  a6,-(a7)
                move.l  IO_DEVICE(a1),a6 ;Pointer to device
                jsr     DEV_BEGINIO(a6)  ;Jump to device's BEGINIO vector
                move.l  (a7)+,a6
                rts

                END



****** amiga.lib/NewList ************************

        INCLUDE "exec/types.i"
        INCLUDE "exec/lists.i"

                SECTION _NewList
                XDEF    _NewList

_NewList:       move.l  4(sp),a0        ;Get pointer from C's stack
                move.l  a0,d0           :pass the list back in D0

;This next code is equavalent to the NEWLIST macro
                clr.l   LH_TAIL(a0)
                move.l  a0,LH_TAILPRED(a0)
                addq.l  #LH_TAIL,a0     ;pointer plus 4...
                move.l  a0,-(a0)        ;...back down to LH_HEAD
                rts

                END
```

C language include files

This section contains the C-language include files from the Amiga operating system source code. These include files define the data structures and constants used by the system software. Whenever the system requires a certain structure or constant, it will be defined in an include file. These include files are organized on a functional basis. For example, files pertinent to the graphics library are listed under graphics/filename.h.

This section is for easy reference only. Similar include files generally come on disk with whatever C compiler you may choose to use with the Amiga. It is impotant to keep in mind that the include files that come with the compiler may not be an exact match to the files listed here.

WARNING: Not all information in this section should be used in your programs. The include files contain definitions for some structure members and constants that are not supported for use by programs. Many of these are marked as private in the comment field of the include file.

Listed below is a simple example of a C program which uses the system header file dos.h.

```
/*
*  A quick example of using a C language include file.  The constant
* "ID_KICKSTART_DISK" is not defined in this example; the value
* is pulled from the "libraries/dos.h" include file.
*/

#include "libraries/dos.h"
void main()
{
      printf("ID_KICKSTART_DISK equals %lx\n",ID_KICKSTART_DISK);
      exit(RETURN_OK);
}
```

clib/alib_protos.h

Page 1

```
#ifndef  CLIB_ALIB_PROTOS_H
#define  CLIB_ALIB_PROTOS_H
/*
**      $Filename: clib/alib_protos.h $
**      $Release: 2.04 $
**      $Revision: 1.1 $
**      $Date: 90/11/07 $
**
**      C prototypes. For use with 32 bit integers only.
**
**      (C) Copyright 1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/
#ifndef  EXEC_TYPES_H
#include <exec/types.h>
#endif
#ifndef  DEVICES_TIMER_H
#include <devices/timer.h>
#endif
#ifndef  LIBRARIES_MATHFFP_H
#include <libraries/mathffp.h>
#endif
#ifndef  LIBRARIES_COMMODITIES_H
#include <libraries/commodities.h>
#endif
/*--- functions in V36 or higher (distributed as Release 2.0) ---*/
void BeginIO( struct IORequest *io );
struct IORequest *CreateExtIO( struct MsgPort *msg, long size );
struct MsgPort *CreatePort( UBYTE *name, long pri );
struct IOStdReq *CreateStdIO( struct MsgPort *msg );
struct Task *CreateTask( UBYTE *name, long pri, APTR initPC,
        unsigned long stackSize );
void DeleteExtIO( struct IORequest *io );
void DeletePort( struct MsgPort *io );
void DeleteStdIO( struct IOStdReq *io );
void DeleteTask( struct Task *task );
void NewList( struct List *list );
LONG NameFromAnchor( struct AnchorPath *anchor, UBYTE *buffer, long buflen );
/* in clib; from graphics.library; */
void AddTOF( struct Isrvstr *i, long (*p)(), long a );
void RemTOF( struct Isrvstr *i );
void waitbeam( long b );
/* in math_support */
FLOAT afp( BYTE *string );
void arnd( long place, long exp, BYTE *string );
FLOAT dbf( unsigned long exp, unsigned long mant );
LONG fpa( FLOAT fnum, BYTE *string );
void fpbcd( FLOAT fnum, BYTE *string );
/* in timer_support */
LONG TimeDelay( long unit, unsigned long secs, unsigned long microsecs );
LONG DoTimer( struct timeval *, long unit, long command );
/**/
/*  Amiga.lib Functions */
/**/
void ArgArrayDone( void );
UBYTE **ArgArrayInit( long arg1, UBYTE **arg2 );
LONG ArgInt( UBYTE **arg1, UBYTE *arg2, long arg3 );
UBYTE *ArgString( UBYTE **arg1, UBYTE *arg2, UBYTE *arg3 );
CxObj *HotKey( UBYTE *arg1, struct MsgPort *arg2, long arg3 );
struct InputEvent *InvertString( UBYTE *arg1, ULONG *arg2 );
/**/
/* Macros */
/**/
/* CxObj  *CxCustom(LONG(*)(),LONG)                   (A0,D0) */
/* CxObj  *CxDebug(LONG)                              (D0) */
/* CxObj  *CxFilter(BYTE *)                           (A0) */
```

clib/alib_protos.h

Page 2

```
/* CxObj  *CxSender(struct MsgPort *,LONG)            (A0,D0) */
/* CxObj  *CxSignal(struct Task *,LONG)           (A0,D0) */
/* CxObj  *CxTranslate(struct InputEvent *)           (A0) */
/* CxObj  *CxTypeFilter(LONG)                         (D0) */
#endif   /* CLIB_ALIB_PROTOS_H */
```

clib/alib_stdio_protos.h

```
#ifndef  CLIB_ALIB_STDIO_PROTOS_H
#define  CLIB_ALIB_STDIO_PROTOS_H
/*
**      $Filename: clib/alib_stdio_protos.h $
**      $Release: 2.04 $
**      $Revision: 1.1 $
**      $Date: 90/11/06 $
**
**      C prototypes. For use with 32 bit integers only.
**
**      (C) Copyright 1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/
#ifndef  EXEC_TYPES_H
#include <exec/types.h>
#endif
/* functions that duplicate those in a typical C library */
LONG printf( STRPTR fmt, ... );
LONG sprintf( STRPTR buffer, STRPTR fmt, ... );
LONG fclose( long stream );
LONG fgetc( long stream );
LONG fprintf( long stream, STRPTR fmt, ... );
LONG fputc( long c, long stream );
LONG fputs( UBYTE *s, long stream );
LONG getchar( void );
LONG putchar( long c );
LONG puts( BYTE *s );
#endif   /* CLIB_ALIB_STDIO_PROTOS_H */
```

clib/all_protos.h

```
#include        <clib/alib_protos.h>
#include        <clib/alib_stdio_protos.h>
#include        <clib/asl_protos.h>
#include        <clib/battclock_protos.h>
#include        <clib/battmem_protos.h>
#include        <clib/cia_protos.h>
#include        <clib/commodities_protos.h>
#include        <clib/console_protos.h>
#include        <clib/diskfont_protos.h>
#include        <clib/dos_protos.h>
#include        <clib/exec_protos.h>
#include        <clib/expansion_protos.h>
#include        <clib/gadtools_protos.h>
#include        <clib/graphics_protos.h>
#include        <clib/icon_protos.h>
#include        <clib/iffparse_protos.h>
#include        <clib/input_protos.h>
#include        <clib/intuition_protos.h>
#include        <clib/keymap_protos.h>
#include        <clib/layers_protos.h>
#include        <clib/macros.h>
#include        <clib/mathffp_protos.h>
#include        <clib/mathieeedoubbas_protos.h>
#include        <clib/mathieeedoubtrans_protos.h>
#include        <clib/mathieeesingbas_protos.h>
#include        <clib/mathieeesingtrans_protos.h>
#include        <clib/mathtrans_protos.h>
#include        <clib/misc_protos.h>
#include        <clib/potgo_protos.h>
#include        <clib/ramdrive_protos.h>
#include        <clib/timer_protos.h>
#include        <clib/translator_protos.h>
#include        <clib/utility_protos.h>
#include        <clib/wb_protos.h>
```

clib/asl_protos.h

Page 1

```
#ifndef  CLIB_ASL_PROTOS_H
#define  CLIB_ASL_PROTOS_H
/*
**      $Filename: clib/asl_protos.h $
**      $Release: 2.04 $
**      $Revision: 36.2 $
**      $Date: 91/01/23 $
**
**      C prototypes. For use with 32 bit integers only.
**
**      (C) Copyright 1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/
/* "asl.library" */
#ifndef  EXEC_TYPES_H
#include <exec/types.h>
#endif
#ifndef  UTILITY_TAGITEM_H
#include <utility/tagitem.h>
#endif
#ifndef  LIBRARIES_ASL_H
#include <libraries/asl.h>
#endif
/*--- functions in V36 or higher (distributed as Release 2.0) ---*/
/**/
struct FileRequester *AllocFileRequest( void );
void FreeFileRequest( struct FileRequester *fileReq );
BOOL RequestFile( struct FileRequester *fileReq );
APTR AllocAslRequest( unsigned long type, struct TagItem *tagList );
APTR AllocAslRequestTags( unsigned long type, Tag Tag1, ... );
void FreeAslRequest( APTR request );
BOOL AslRequest( APTR request, struct TagItem *tagList );
BOOL AslRequestTags( APTR request, Tag Tag1, ... );
#endif   /* CLIB_ASL_PROTOS_H */
```

clib/battclock_protos.h

Page 1

```
#ifndef  CLIB_BATTCLOCK_PROTOS_H
#define  CLIB_BATTCLOCK_PROTOS_H
/*
**      $Filename: clib/battclock_protos.h $
**      $Release: 2.04 $
**      $Revision: 1.3 $
**      $Date: 90/05/03 $
**
**      C prototypes. For use with 32 bit integers only.
**
**      (C) Copyright 1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/
/* "battclock.resource" */
#ifndef  EXEC_TYPES_H
#include <exec/types.h>
#endif
void ResetBattClock( void );
ULONG ReadBattClock( void );
void WriteBattClock( unsigned long time );
#endif   /* CLIB_BATTCLOCK_PROTOS_H */
```

clib/battmem_protos.h

```
#ifndef  CLIB_BATTMEM_PROTOS_H
#define  CLIB_BATTMEM_PROTOS_H
/*
**      $Filename: clib/battmem_protos.h $
**      $Release: 2.04 $
**      $Revision: 1.5 $
**      $Date: 91/03/04 $
**
**      C prototypes. For use with 32 bit integers only.
**
**      (C) Copyright 1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/
/* "battmem.resource" */
#ifndef  EXEC_TYPES_H
#include <exec/types.h>
#endif
void ObtainBattSemaphore( void );
void ReleaseBattSemaphore( void );
ULONG ReadBattMem( APTR buffer, unsigned long offset, unsigned long length );
ULONG WriteBattMem( APTR buffer, unsigned long offset, unsigned long length );
#endif   /* CLIB_BATTMEM_PROTOS_H */
```

clib/cia_protos.h

```
#ifndef  CLIB_CIA_PROTOS_H
#define  CLIB_CIA_PROTOS_H
/*
**      $Filename: clib/cia_protos.h $
**      $Release: 2.04 $
**      $Revision: 1.7 $
**      $Date: 90/07/19 $
**
**      C prototypes. For use with 32 bit integers only.
**
**      (C) Copyright 1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/
/* "CiaA.Resource" and "CiaB.Resource" */
#ifndef  EXEC_TYPES_H
#include <exec/types.h>
#endif
#ifndef  EXEC_INTERRUPTS_H
#include <exec/interrupts.h>
#endif
#ifndef  EXEC_LIBRARIES_H
#include <exec/libraries.h>
#endif
struct Interrupt *AddICRVector( struct Library *resource, long iCRBit,
        struct Interrupt *interrupt );
void RemICRVector( struct Library *resource, long iCRBit,
        struct Interrupt *interrupt );
WORD AbleICR( struct Library *resource, long mask );
WORD SetICR( struct Library *resource, long mask );
#endif   /* CLIB_CIA_PROTOS_H */
```

clib/commodities_protos.h Page 1

```
#ifndef  CLIB_COMMODITIES_PROTOS_H
#define  CLIB_COMMODITIES_PROTOS_H
/*
**      $Filename: clib/commodities_protos.h $
**      $Release: 2.04 $
**      $Revision: 37.1 $
**      $Date: 91/02/19 $
**
**      C prototypes. For use with 32 bit integers only.
**
**      (C) Copyright 1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/
/* "commodities.library" */
#ifndef  EXEC_TYPES_H
#include <exec/types.h>
#endif
#ifndef  EXEC_NODES_H
#include <exec/nodes.h>
#endif
#ifndef  LIBRARIES_COMMODITIES_H
#include <libraries/commodities.h>
#endif
#ifndef  DEVICES_INPUTEVENT_H
#include <devices/inputevent.h>
#endif
#ifndef  DEVICES_KEYMAP_H
#include <devices/keymap.h>
#endif
/*--- functions in V36 or higher (distributed as Release 2.0) ---*/
/**/
/*  OBJECT UTILITIES */
/**/
CxObj *CreateCxObj( unsigned long type, long arg1, long arg2 );
CxObj *CxBroker( struct NewBroker *nb, LONG *error );
LONG ActivateCxObj( CxObj *co, long true );
void DeleteCxObj( CxObj *co );
void DeleteCxObjAll( CxObj *co );
ULONG CxObjType( CxObj *co );
LONG CxObjError( CxObj *co );
void ClearCxObjError( CxObj *co );
void SetCxObjPri( CxObj *co, long pri );
/**/
/*  OBJECT ATTACHMENT */
/**/
void AttachCxObj( CxObj *headobj, CxObj *co );
void EnqueueCxObj( CxObj *headobj, CxObj *co );
void InsertCxObj( CxObj *headobj, CxObj *co, CxObj *pred );
void RemoveCxObj( CxObj *co );
/**/
/*  TYPE SPECIFIC */
/**/
void SetTranslate( CxObj *translator, IX *ie );
void SetFilter( CxObj *filter, IX *text );
void SetFilterIX( CxObj *filter, IX *ix );
LONG ParseIX( UBYTE *descr, IX *ix );
/**/
/*  COMMON MESSAGE */
/**/
ULONG CxMsgType( CxMsg *cxm );
UBYTE *CxMsgData( CxMsg *cxm );
LONG CxMsgID( CxMsg *cxm );
/**/
/*  MESSAGE ROUTING */
/**/
void DivertCxMsg( CxMsg *cxm, CxObj *headobj, CxObj *ret );
```

clib/commodities_protos.h Page 2

```
void RouteCxMsg( CxMsg *cxm, CxObj *co );
void DisposeCxMsg( CxMsg *cxm );
/**/
/*  INPUT EVENT HANDLING */
/**/
ULONG InvertKeyMap( unsigned long ansicode, struct InputEvent *ie,
        struct KeyMap *km );
void AddIEvents( struct InputEvent *ie );
/**/
/* FOR USE ONLY BY CONTROLLER PROGRAM */
/**/
#endif   /* CLIB_COMMODITIES_PROTOS_H */
```

clib/console_protos.h

Page 1

```
#ifndef  CLIB_CONSOLE_PROTOS_H
#define  CLIB_CONSOLE_PROTOS_H
/*
**      $Filename: clib/console_protos.h $
**      $Release: 2.04 $
**      $Revision: 36.6 $
**      $Date: 90/11/07 $
**
**      C prototypes. For use with 32 bit integers only.
**
**      (C) Copyright 1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/
/* "console.device" */
#ifndef  EXEC_LIBRARIES_H
#include <exec/libraries.h>
#endif
#ifndef  DEVICES_INPUTEVENT_H
#include <devices/inputevent.h>
#endif
#ifndef  DEVICES_KEYMAP_H
#include <devices/keymap.h>
#endif
struct InputEvent *CDInputHandler( struct InputEvent *events,
        struct Library *consoleDevice );
LONG RawKeyConvert( struct InputEvent *events, STRPTR buffer, long length,
        struct KeyMap *keyMap );
/*--- functions in V36 or higher (distributed as Release 2.0) ---*/
#endif   /* CLIB_CONSOLE_PROTOS_H */
```

clib/disk_protos.h

Page 1

```
#ifndef  CLIB_DISK_PROTOS_H
#define  CLIB_DISK_PROTOS_H
/*
**      $Filename: clib/disk_protos.h $
**      $Release: 2.04 $
**      $Revision: 36.1 $
**      $Date: 91/02/19 $
**
**      C prototypes. For use with 32 bit integers only.
**
**      (C) Copyright 1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/
/* "disk.resource" */
#ifndef  RESOURCES_DISK_H
#include <resources/disk.h>
#endif
BOOL AllocUnit( long unitNum );
void FreeUnit( long unitNum );
struct DiskResourceUnit *GetUnit( struct DiskResourceUnit *unitPointer );
void GiveUnit( void );
LONG GetUnitID( long unitNum );
/*------ new for V37 ------*/
LONG ReadUnitID( long unitNum );
#endif   /* CLIB_DISK_PROTOS_H */
```

clib/diskfont_protos.h

```
#ifndef  CLIB_DISKFONT_PROTOS_H
#define  CLIB_DISKFONT_PROTOS_H
/*
**      $Filename: clib/diskfont_protos.h $
**      $Release: 2.04 $
**      $Revision: 36.2 $
**      $Date: 90/07/19 $
**
**      C prototypes. For use with 32 bit integers only.
**
**      (C) Copyright 1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/
/* "diskfont.library" */
#ifndef  DOS_DOS_H
#include <dos/dos.h>
#endif
#ifndef  LIBRARIES_DISKFONT_H
#include <libraries/diskfont.h>
#endif
struct TextFont *OpenDiskFont( struct TextAttr *textAttr );
LONG AvailFonts( STRPTR buffer, long bufBytes, long flags );
/*--- functions in V34 or higher (distributed as Release 1.3) ---*/
struct FontContentsHeader *NewFontContents( BPTR fontsLock, STRPTR fontName );
void DisposeFontContents( struct FontContentsHeader *fontContentsHeader );
/*--- functions in V36 or higher (distributed as Release 2.0) ---*/
struct DiskFontHeader *NewScaledDiskFont( struct TextFont *sourceFont,
        struct TextAttr *destTextAttr );
#endif   /* CLIB_DISKFONT_PROTOS_H */
```

clib/dos_protos.h

```
#ifndef  CLIB_DOS_PROTOS_H
#define  CLIB_DOS_PROTOS_H
/*
**      $Filename: clib/dos_protos.h $
**      $Release: 2.04 $
**      $Revision: 36.21 $
**      $Date: 91/02/19 $
**
**      C prototypes. For use with 32 bit integers only.
**
**      (C) Copyright 1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/
/* "dos.library" */
#ifndef  DOS_DOS_H
#include <dos/dos.h>
#endif
#ifndef  DOS_DOSEXTENS_H
#include <dos/dosextens.h>
#endif
#ifndef  DOS_RECORD_H
#include <dos/record.h>
#endif
#ifndef  DOS_RDARGS_H
#include <dos/rdargs.h>
#endif
#ifndef  DOS_DOSASL_H
#include <dos/dosasl.h>
#endif
#ifndef  DOS_VAR_H
#include <dos/var.h>
#endif
#ifndef  DOS_NOTIFY_H
#include <dos/notify.h>
#endif
#ifndef  DOS_DATETIME_H
#include <dos/datetime.h>
#endif
BPTR Open( UBYTE *name, long accessMode );
LONG Close( BPTR file );
LONG Read( BPTR file, APTR buffer, long length );
LONG Write( BPTR file, APTR buffer, long length );
BPTR Input( void );
BPTR Output( void );
LONG Seek( BPTR file, long position, long offset );
LONG DeleteFile( UBYTE *name );
LONG Rename( UBYTE *oldName, UBYTE *newName );
BPTR Lock( UBYTE *name, long type );
void UnLock( BPTR lock );
BPTR DupLock( BPTR lock );
LONG Examine( BPTR lock, struct FileInfoBlock *fileInfoBlock );
LONG ExNext( BPTR lock, struct FileInfoBlock *fileInfoBlock );
LONG Info( BPTR lock, struct InfoData *parameterBlock );
BPTR CreateDir( UBYTE *name );
BPTR CurrentDir( BPTR lock );
LONG IoErr( void );
struct MsgPort *CreateProc( UBYTE *name, long pri, BPTR segList,
        long stackSize );
void Exit( long returnCode );
BPTR LoadSeg( UBYTE *name );
void UnLoadSeg( BPTR seglist );
struct MsgPort *DeviceProc( UBYTE *name );
LONG SetComment( UBYTE *name, UBYTE *comment );
LONG SetProtection( UBYTE *name, long protect );
struct DateStamp *DateStamp( struct DateStamp *date );
void Delay( long timeout );
```

```
LONG WaitForChar( BPTR file, long timeout );
BPTR ParentDir( BPTR lock );
LONG IsInteractive( BPTR file );
LONG Execute( UBYTE *string, BPTR file, BPTR file2 );
/*--- functions in V36 or higher (distributed as Release 2.0) ---*/
/*      DOS Object creation/deletion */
APTR AllocDosObject( unsigned long type, struct TagItem *tags );
APTR AllocDosObjectTagList( unsigned long type, struct TagItem *tags );
APTR AllocDosObjectTags( unsigned long type, unsigned long tag1type, ... );
void FreeDosObject( unsigned long type, APTR ptr );
/*      Packet Level routines */
LONG DoPkt( struct MsgPort *port, long action, long arg1, long arg2, long arg3,
        long arg4, long arg5 );
LONG DoPkt0( struct MsgPort *port, long action );
LONG DoPkt1( struct MsgPort *port, long action, long arg1 );
LONG DoPkt2( struct MsgPort *port, long action, long arg1, long arg2 );
LONG DoPkt3( struct MsgPort *port, long action, long arg1, long arg2,
        long arg3 );
LONG DoPkt4( struct MsgPort *port, long action, long arg1, long arg2,
        long arg3, long arg4 );
void SendPkt( struct DosPacket *dp, struct MsgPort *port,
        struct MsgPort *replyport );
struct DosPacket *WaitPkt( void );
void ReplyPkt( struct DosPacket *dp, long res1, long res2 );
void AbortPkt( struct MsgPort *port, struct DosPacket *pkt );
/*      Record Locking */
BOOL LockRecord( BPTR fh, unsigned long offset, unsigned long length,
        unsigned long mode, unsigned long timeout );
BOOL LockRecords( struct RecordLock *recArray, unsigned long timeout );
BOOL UnLockRecord( BPTR fh, unsigned long offset, unsigned long length );
BOOL UnLockRecords( struct RecordLock *recArray );
/*      Buffered File I/O */
BPTR SelectInput( BPTR fh );
BPTR SelectOutput( BPTR fh );
LONG FGetC( BPTR fh );
void FPutC( BPTR fh, unsigned long ch );
LONG UnGetC( BPTR fh, long character );
LONG FRead( BPTR fh, APTR block, unsigned long blocklen,
        unsigned long number );
LONG FWrite( BPTR fh, APTR block, unsigned long blocklen,
        unsigned long number );
UBYTE *FGets( BPTR fh, UBYTE *buf, unsigned long buflen );
LONG FPuts( BPTR fh, UBYTE *str );
void VFWritef( BPTR fh, UBYTE *format, LONG *argarray );
void FWritef( BPTR fh, UBYTE *format, long arg1, ... );
LONG VFPrintf( BPTR fh, UBYTE *format, LONG *argarray );
LONG FPrintf( BPTR fh, UBYTE *format, long arg1, ... );
void Flush( BPTR fh );
LONG SetVBuf( BPTR fh, UBYTE *buff, long type, long size );
/*      DOS Object Management */
BPTR DupLockFromFH( BPTR fh );
BPTR OpenFromLock( BPTR lock );
BPTR ParentOfFH( BPTR fh );
BOOL ExamineFH( BPTR fh, struct FileInfoBlock *fib );
LONG SetFileDate( UBYTE *name, struct DateStamp *date );
LONG NameFromLock( BPTR lock, UBYTE *buffer, long len );
LONG NameFromFH( BPTR fh, UBYTE *buffer, long len );
WORD SplitName( UBYTE *name, unsigned long seperator, UBYTE *buf, long oldpos,
        long size );
LONG SameLock( BPTR lock1, BPTR lock2 );
LONG SetMode( BPTR fh, long mode );
LONG ExAll( BPTR lock, struct ExAllData *buffer, long size, long data,
        struct ExAllControl *control );
LONG ReadLink( struct MsgPort *port, BPTR lock, UBYTE *path, UBYTE *buffer,
        unsigned long size );
LONG MakeLink( UBYTE *name, long dest, long soft );
```

```
LONG ChangeMode( long type, BPTR fh, long newmode );
LONG SetFileSize( BPTR fh, long pos, long mode );
/*      Error Handling */
LONG SetIoErr( long result );
BOOL Fault( long code, UBYTE *header, UBYTE *buffer, long len );
BOOL PrintFault( long code, UBYTE *header );
LONG ErrorReport( long code, long type, unsigned long arg1,
        struct MsgPort *device );
/*      Process Management */
struct CommandLineInterface *Cli( void );
struct Process *CreateNewProc( struct TagItem *tags );
struct Process *CreateNewProcTagList( struct TagItem *tags );
struct Process *CreateNewProcTags( unsigned long tag1type, ... );
LONG RunCommand( BPTR seg, long stack, UBYTE *paramptr, long paramlen );
struct MsgPort *GetConsoleTask( void );
struct MsgPort *SetConsoleTask( struct MsgPort *task );
struct MsgPort *GetFileSysTask( void );
struct MsgPort *SetFileSysTask( struct MsgPort *task );
UBYTE *GetArgStr( void );
BOOL SetArgStr( UBYTE *string );
struct Process *FindCliProc( unsigned long num );
ULONG MaxCli( void );
BOOL SetCurrentDirName( UBYTE *name );
BOOL GetCurrentDirName( UBYTE *buf, long len );
BOOL SetProgramName( UBYTE *name );
BOOL GetProgramName( UBYTE *buf, long len );
BOOL SetPrompt( UBYTE *name );
BOOL GetPrompt( UBYTE *buf, long len );
BPTR SetProgramDir( BPTR lock );
BPTR GetProgramDir( void );
/*      Device List Management */
LONG SystemTagList( UBYTE *command, struct TagItem *tags );
LONG System( UBYTE *command, struct TagItem *tags );
LONG SystemTags( UBYTE *command, unsigned long tag1type, ... );
LONG AssignLock( UBYTE *name, BPTR lock );
BOOL AssignLate( UBYTE *name, UBYTE *path );
BOOL AssignPath( UBYTE *name, UBYTE *path );
BOOL AssignAdd( UBYTE *name, BPTR lock );
LONG RemAssignList( UBYTE *name, BPTR lock );
struct DevProc *GetDeviceProc( UBYTE *name, struct DevProc *dp );
void FreeDeviceProc( struct DevProc *dp );
struct DosList *LockDosList( unsigned long flags );
void UnLockDosList( unsigned long flags );
struct DosList *AttemptLockDosList( unsigned long flags );
BOOL RemDosEntry( struct DosList *dlist );
struct DosList *AddDosEntry( struct DosList *dlist );
struct DosList *FindDosEntry( struct DosList *dlist, UBYTE *name,
        unsigned long flags );
struct DosList *NextDosEntry( struct DosList *dlist, unsigned long flags );
struct DosList *MakeDosEntry( UBYTE *name, long type );
void FreeDosEntry( struct DosList *dlist );
BOOL IsFileSystem( UBYTE *name );
/*      Handler Interface */
BOOL Format( UBYTE *filesystem, UBYTE *volumename, unsigned long dostype );
LONG Relabel( UBYTE *drive, UBYTE *newname );
LONG Inhibit( UBYTE *name, long onoff );
LONG AddBuffers( UBYTE *name, long number );
/*      Date, Time Routines */
LONG CompareDates( struct DateStamp *date1, struct DateStamp *date2 );
LONG DateToStr( struct DateTime *datetime );
LONG StrToDate( struct DateTime *datetime );
/*      Image Management */
BPTR InternalLoadSeg( BPTR fh, BPTR table, LONG *funcarray, LONG *stack );
void InternalUnLoadSeg( BPTR seglist, void (*freefunc)() );
BPTR NewLoadSeg( UBYTE *file, struct TagItem *tags );
BPTR NewLoadSegTagList( UBYTE *file, struct TagItem *tags );
```

clib/dos_protos.h

Page 4

```
BPTR NewLoadSegTags( UBYTE *file, unsigned long tagltype, ... );
LONG AddSegment( UBYTE *name, BPTR seg, long system );
struct Segment *FindSegment( UBYTE *name, struct Segment *seg, long system );
LONG RemSegment( struct Segment *seg );
/*      Command Support */
LONG CheckSignal( long mask );
struct RDArgs *ReadArgs( UBYTE *template, LONG *array, struct RDArgs *args );
LONG FindArg( UBYTE *keyword, UBYTE *template );
LONG ReadItem( UBYTE *name, long maxchars, struct CSource *cSource );
LONG StrToLong( UBYTE *string, LONG *value );
LONG MatchFirst( UBYTE *pat, struct AnchorPath *anchor );
LONG MatchNext( struct AnchorPath *anchor );
void MatchEnd( struct AnchorPath *anchor );
BOOL ParsePattern( UBYTE *pat, UBYTE *buf, long buflen );
BOOL MatchPattern( UBYTE *pat, UBYTE *str );
/* Not currently implemented. */
void FreeArgs( struct RDArgs *args );
UBYTE *FilePart( UBYTE *path );
UBYTE *PathPart( UBYTE *path );
BOOL AddPart( UBYTE *dirname, UBYTE *filename, unsigned long size );
/*      Notification */
BOOL StartNotify( struct NotifyRequest *notify );
void EndNotify( struct NotifyRequest *notify );
/*      Environment Variable functions */
BOOL SetVar( UBYTE *name, UBYTE *buffer, long size, long flags );
LONG GetVar( UBYTE *name, UBYTE *buffer, long size, long flags );
LONG DeleteVar( UBYTE *name, unsigned long flags );
struct LocalVar *FindVar( UBYTE *name, unsigned long type );
LONG CliInit( struct DosPacket *dp );
LONG CliInitNewcli( struct DosPacket *dp );
LONG CliInitRun( struct DosPacket *dp );
LONG WriteChars( UBYTE *buf, unsigned long buflen );
LONG PutStr( UBYTE *str );
LONG VPrintf( UBYTE *format, LONG *argarray );
LONG Printf( UBYTE *format, long argl, ... );
/* these were unimplemented until dos 36.147 */
BOOL ParsePatternNoCase( UBYTE *pat, UBYTE *buf, long buflen );
BOOL MatchPatternNoCase( UBYTE *pat, UBYTE *str );
/* this was added for V37 dos, returned 0 before then. */
BOOL SameDevice( BPTR lock1, BPTR lock2 );
/* These were added in dos 36.147 */
/* these were added in dos 37.1 */
/* these were added in dos 37.8 */
#endif   /* CLIB_DOS_PROTOS_H */
```

clib/exec_protos.h

Page 1

```
#ifndef  CLIB_EXEC_PROTOS_H
#define  CLIB_EXEC_PROTOS_H
/*
**      $Filename: clib/exec_protos.h $
**      $Release: 2.04 $
**      $Revision: 36.7 $
**      $Date: 90/11/07 $
**
**      C prototypes. For use with 32 bit integers only.
**
**      (C) Copyright 1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/
#ifndef  EXEC_TYPES_H
#include <exec/types.h>
#endif
/*------ misc ----------------------------------------------------------*/
ULONG Supervisor( unsigned long (*userFunction)() );
/*------ special patchable hooks to internal exec activity -------------*/
/*------ module creation -----------------------------------------------*/
void InitCode( unsigned long startClass, unsigned long version );
void InitStruct( APTR initTable, APTR memory, unsigned long size );
struct Library *MakeLibrary( APTR funcInit, APTR structInit,
        unsigned long (*libInit)(), unsigned long dataSize,
        unsigned long segList );
void MakeFunctions( APTR target, APTR functionArray,
        unsigned long funcDispBase );
struct Resident *FindResident( UBYTE *name );
void InitResident( struct Resident *resident, unsigned long segList );
/*------ diagnostics ---------------------------------------------------*/
void Alert( unsigned long alertNum );
void Debug( unsigned long flags );
/*------ interrupts ----------------------------------------------------*/
void Disable( void );
void Enable( void );
void Forbid( void );
void Permit( void );
ULONG SetSR( unsigned long newSR, unsigned long mask );
APTR SuperState( void );
void UserState( APTR sysStack );
struct Interrupt *SetIntVector( long intNumber, struct Interrupt *interrupt );
void AddIntServer( long intNumber, struct Interrupt *interrupt );
void RemIntServer( long intNumber, struct Interrupt *interrupt );
void Cause( struct Interrupt *interrupt );
/*------ memory allocation ---------------------------------------------*/
APTR Allocate( struct MemHeader *freeList, unsigned long byteSize );
void Deallocate( struct MemHeader *freeList, APTR memoryBlock,
        unsigned long byteSize );
APTR AllocMem( unsigned long byteSize, unsigned long requirements );
APTR AllocAbs( unsigned long byteSize, APTR location );
void FreeMem( APTR memoryBlock, unsigned long byteSize );
ULONG AvailMem( unsigned long requirements );
struct MemList *AllocEntry( struct MemList *entry );
void FreeEntry( struct MemList *entry );
/*------ lists ---------------------------------------------------------*/
void Insert( struct List *list, struct Node *node, struct Node *pred );
void AddHead( struct List *list, struct Node *node );
void AddTail( struct List *list, struct Node *node );
void Remove( struct Node *node );
struct Node *RemHead( struct List *list );
struct Node *RemTail( struct List *list );
void Enqueue( struct List *list, struct Node *node );
struct Node *FindName( struct List *list, UBYTE *name );
/*------ tasks ---------------------------------------------------------*/
APTR AddTask( struct Task *task, APTR initPC, APTR finalPC );
void RemTask( struct Task *task );
```

clib/exec_protos.h Page 2

```
struct Task *FindTask( UBYTE *name );
BYTE SetTaskPri( struct Task *task, long priority );
ULONG SetSignal( unsigned long newSignals, unsigned long signalSet );
ULONG SetExcept( unsigned long newSignals, unsigned long signalSet );
ULONG Wait( unsigned long signalSet );
void Signal( struct Task *task, unsigned long signalSet );
BYTE AllocSignal( long signalNum );
void FreeSignal( long signalNum );
LONG AllocTrap( long trapNum );
void FreeTrap( long trapNum );
/*------ messages -----------------------------------------------------*/
void AddPort( struct MsgPort *port );
void RemPort( struct MsgPort *port );
void PutMsg( struct MsgPort *port, struct Message *message );
struct Message *GetMsg( struct MsgPort *port );
void ReplyMsg( struct Message *message );
struct Message *WaitPort( struct MsgPort *port );
struct MsgPort *FindPort( UBYTE *name );
/*------ libraries ----------------------------------------------------*/
void AddLibrary( struct Library *library );
void RemLibrary( struct Library *library );
struct Library *OldOpenLibrary( UBYTE *libName );
void CloseLibrary( struct Library *library );
APTR SetFunction( struct Library *library, long funcOffset,
        unsigned long (*newFunction)() );
void SumLibrary( struct Library *library );
/*------ devices ------------------------------------------------------*/
void AddDevice( struct Device *device );
void RemDevice( struct Device *device );
BYTE OpenDevice( UBYTE *devName, unsigned long unit,
        struct IORequest *ioRequest, unsigned long flags );
void CloseDevice( struct IORequest *ioRequest );
BYTE DoIO( struct IORequest *ioRequest );
void SendIO( struct IORequest *ioRequest );
BOOL CheckIO( struct IORequest *ioRequest );
BYTE WaitIO( struct IORequest *ioRequest );
void AbortIO( struct IORequest *ioRequest );
/*------ resources ----------------------------------------------------*/
void AddResource( APTR resource );
void RemResource( APTR resource );
APTR OpenResource( UBYTE *resName );
/*------ private diagnostic support -----------------------------------*/
/*------ misc ---------------------------------------------------------*/
void RawDoFmt( UBYTE *formatString, APTR dataStream, void (*putChProc)(),
        APTR putChData );
ULONG GetCC( void );
ULONG TypeOfMem( APTR address );
ULONG Procure( struct Semaphore *semaport, struct Message *bidMsg );
void Vacate( struct Semaphore *semaport );
struct Library *OpenLibrary( UBYTE *libName, unsigned long version );
/*--- functions in V33 or higher (distributed as Release 1.2) ---*/
/*------ signal semaphores (note funny registers)----------------------*/
void InitSemaphore( struct SignalSemaphore *sigSem );
void ObtainSemaphore( struct SignalSemaphore *sigSem );
void ReleaseSemaphore( struct SignalSemaphore *sigSem );
ULONG AttemptSemaphore( struct SignalSemaphore *sigSem );
void ObtainSemaphoreList( struct List *sigSem );
void ReleaseSemaphoreList( struct List *sigSem );
struct SignalSemaphore *FindSemaphore( UBYTE *sigSem );
void AddSemaphore( struct SignalSemaphore *sigSem );
void RemSemaphore( struct SignalSemaphore *sigSem );
/*------ kickmem support ----------------------------------------------*/
ULONG SumKickData( void );
/*------ more memory support ------------------------------------------*/
ULONG AddMemList( unsigned long size, unsigned long attributes,
        unsigned long pri, APTR base, UBYTE *name );
```

clib/exec_protos.h Page 3

```
void CopyMem( APTR source, APTR dest, unsigned long size );
void CopyMemQuick( APTR source, APTR dest, unsigned long size );
/*------ cache --------------------------------------------------------*/
/*--- functions in V36 or higher (distributed as Release 2.0) ---*/
void CacheClearU( void );
void CacheClearE( APTR address, unsigned long length, unsigned long caches );
ULONG CacheControl( unsigned long cacheBits, unsigned long cacheMask );
/*------ misc ---------------------------------------------------------*/
APTR CreateIORequest( struct MsgPort *port, unsigned long size );
void DeleteIORequest( APTR iorequest );
struct MsgPort *CreateMsgPort( void );
void DeleteMsgPort( struct MsgPort *port );
void ObtainSemaphoreShared( struct SignalSemaphore *sigSem );
/*------ even more memory support -------------------------------------*/
APTR AllocVec( unsigned long byteSize, unsigned long requirements );
void FreeVec( APTR memoryBlock );
APTR CreatePrivatePool( unsigned long requirements, unsigned long puddleSize,
        unsigned long puddleThresh );
void DeletePrivatePool( APTR poolHeader );
APTR AllocPooled( unsigned long memSize, APTR poolHeader );
void FreePooled( APTR memory, APTR poolHeader );
/*------ misc ---------------------------------------------------------*/
void ColdReboot( void );
void StackSwap( APTR newSize, APTR newSP, APTR newStack );
/*------ task trees ---------------------------------------------------*/
void ChildFree( APTR tid );
void ChildOrphan( APTR tid );
void ChildStatus( APTR tid );
void ChildWait( APTR tid );
/*------ future expansion ---------------------------------------------*/
#endif  /* CLIB_EXEC_PROTOS_H */
```

clib/expansion_protos.h

```
#ifndef  CLIB_EXPANSION_PROTOS_H
#define  CLIB_EXPANSION_PROTOS_H
/*
**      $Filename: clib/expansion_protos.h $
**      $Release: 2.04 $
**      $Revision: 36.6 $
**      $Date: 91/02/08 $
**
**      C prototypes. For use with 32 bit integers only.
**
**      (C) Copyright 1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/
#ifndef  EXEC_TYPES_H
#include <exec/types.h>
#endif
/*--- functions in V33 or higher (distributed as Release 1.2) ---*/
void AddConfigDev( struct ConfigDev *configDev );
/*--- functions in V36 or higher (distributed as Release 2.0) ---*/
BOOL AddBootNode( long bootPri, unsigned long flags,
        struct DeviceNode *deviceNode, struct ConfigDev *configDev );
/*--- functions in V33 or higher (distributed as Release 1.2) ---*/
void AllocBoardMem( unsigned long slotSpec );
struct ConfigDev *AllocConfigDev( void );
APTR AllocExpansionMem( unsigned long numSlots, unsigned long slotAlign );
void ConfigBoard( APTR board, struct ConfigDev *configDev );
void ConfigChain( APTR baseAddr );
struct ConfigDev *FindConfigDev( struct ConfigDev *oldConfigDev,
        long manufacturer, long product );
void FreeBoardMem( unsigned long startSlot, unsigned long slotSpec );
void FreeConfigDev( struct ConfigDev *configDev );
void FreeExpansionMem( unsigned long startSlot, unsigned long numSlots );
UBYTE ReadExpansionByte( APTR board, unsigned long offset );
void ReadExpansionRom( APTR board, struct ConfigDev *configDev );
void RemConfigDev( struct ConfigDev *configDev );
void WriteExpansionByte( APTR board, unsigned long offset,
        unsigned long byte );
void ObtainConfigBinding( void );
void ReleaseConfigBinding( void );
void SetCurrentBinding( struct CurrentBinding *currentBinding,
        unsigned long bindingSize );
ULONG GetCurrentBinding( struct CurrentBinding *currentBinding,
        unsigned long bindingSize );
struct DeviceNode *MakeDosNode( APTR parmPacket );
BOOL AddDosNode( long bootPri, unsigned long flags,
        struct DeviceNode *deviceNode );
/*--- functions in V36 or higher (distributed as Release 2.0) ---*/
#endif   /* CLIB_EXPANSION_PROTOS_H */
```

clib/gadtools_protos.h

```
#ifndef  CLIB_GADTOOLS_PROTOS_H
#define  CLIB_GADTOOLS_PROTOS_H
/*
**      $Filename: clib/gadtools_protos.h $
**      $Release: 2.04 $
**      $Revision: 36.2 $
**      $Date: 90/05/02 $
**
**      C prototypes. For use with 32 bit integers only.
**
**      (C) Copyright 1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/
/* "gadtools.library" */
#ifndef  EXEC_TYPES_H
#include <exec/types.h>
#endif
#ifndef  INTUITION_INTUITION_H
#include <intuition/intuition.h>
#endif
#ifndef  UTILITY_TAGITEM_H
#include <utility/tagitem.h>
#endif
#ifndef  LIBRARIES_GADTOOLS_H
#include <libraries/gadtools.h>
#endif
/*--- functions in V36 or higher (distributed as Release 2.0) ---*/
/**/
/* Gadget Functions */
/**/
struct Gadget *CreateGadgetA( unsigned long kind, struct Gadget *gad,
        struct NewGadget *ng, struct TagItem *taglist );
struct Gadget *CreateGadget( unsigned long kind, struct Gadget *gad,
        struct NewGadget *ng, Tag tag1, ... );
void FreeGadgets( struct Gadget *gad );
void GT_SetGadgetAttrsA( struct Gadget *gad, struct Window *win,
        struct Requester *req, struct TagItem *taglist );
void GT_SetGadgetAttrs( struct Gadget *gad, struct Window *win,
        struct Requester *req, Tag tag1, ... );
/**/
/* Menu functions */
/**/
struct Menu *CreateMenusA( struct NewMenu *newmenu, struct TagItem *taglist );
struct Menu *CreateMenus( struct NewMenu *newmenu, Tag tag1, ... );
void FreeMenus( struct Menu *menu );
BOOL LayoutMenuItemsA( struct MenuItem *firstitem, APTR vi,
        struct TagItem *taglist );
BOOL LayoutMenuItems( struct MenuItem *firstitem, APTR vi, Tag tag1, ... );
BOOL LayoutMenusA( struct Menu *firstmenu, APTR vi, struct TagItem *taglist );
BOOL LayoutMenus( struct Menu *firstmenu, APTR vi, Tag tag1, ... );
/**/
/* Misc Event-Handling Functions */
/**/
struct IntuiMessage *GT_GetIMsg( struct MsgPort *iport );
void GT_ReplyIMsg( struct IntuiMessage *imsg );
void GT_RefreshWindow( struct Window *win, struct Requester *req );
void GT_BeginRefresh( struct Window *win );
void GT_EndRefresh( struct Window *win, long complete );
struct IntuiMessage *GT_FilterIMsg( struct IntuiMessage *imsg );
struct IntuiMessage *GT_PostFilterIMsg( struct IntuiMessage *imsg );
struct Gadget *CreateContext( struct Gadget **glistptr );
/**/
/* Rendering Functions */
/**/
void DrawBevelBoxA( struct RastPort *rport, long left, long top, long width,
        long height, struct TagItem *taglist );
```

clib/gadtools_protos.h Page 2

```c
void DrawBevelBox( struct RastPort *rport, long left, long top, long width,
        long height, Tag tag1, ... );
/**/
/* Visuals Functions */
/**/
APTR GetVisualInfoA( struct Screen *screen, struct TagItem *taglist );
APTR GetVisualInfo( struct Screen *screen, Tag tag1, ... );
void FreeVisualInfo( APTR vi );
/**/
/* Reserved entries */
/**/
#endif   /* CLIB_GADTOOLS_PROTOS_H */
```

clib/graphics_protos.h Page 1

```c
#ifndef  CLIB_GRAPHICS_PROTOS_H
#define  CLIB_GRAPHICS_PROTOS_H
/*
**      $Filename: clib/graphics_protos.h $
**      $Release: 2.04 $
**      $Revision: 37.0 $
**      $Date: 91/01/07 $
**
**      C prototypes. For use with 32 bit integers only.
**
**      (C) Copyright 1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/
/* "graphics.library" */
#ifndef  GRAPHICS_GFX_H
#include <graphics/gfx.h>
#endif
#ifndef  GRAPHICS_DISPLAYINFO_H
#include <graphics/displayinfo.h>
#endif
#ifndef  GRAPHICS_GELS_H
#include <graphics/gels.h>
#endif
#ifndef  GRAPHICS_RASTPORT_H
#include <graphics/rastport.h>
#endif
#ifndef  GRAPHICS_VIEW_H
#include <graphics/view.h>
#endif
#ifndef  GRAPHICS_COPPER_H
#include <graphics/copper.h>
#endif
#ifndef  GRAPHICS_CLIP_H
#include <graphics/clip.h>
#endif
#ifndef  GRAPHICS_REGIONS_H
#include <graphics/regions.h>
#endif
#ifndef  GRAPHICS_SPRITE_H
#include <graphics/sprite.h>
#endif
#ifndef  GRAPHICS_TEXT_H
#include <graphics/text.h>
#endif
#ifndef  HARDWARE_BLIT_H
#include <hardware/blit.h>
#endif
/*------ BitMap primitives ------*/
LONG BltBitMap( struct BitMap *srcBitMap, long xSrc, long ySrc,
        struct BitMap *destBitMap, long xDest, long yDest, long xSize,
        long ySize, unsigned long minterm, unsigned long mask,
        PLANEPTR tempA );
void BltTemplate( PLANEPTR source, long xSrc, long srcMod,
        struct RastPort *destRP, long xDest, long yDest, long xSize,
        long ySize );
/*------ Text routines ------*/
void ClearEOL( struct RastPort *rp );
void ClearScreen( struct RastPort *rp );
WORD TextLength( struct RastPort *rp, STRPTR string, unsigned long count );
LONG Text( struct RastPort *rp, STRPTR string, unsigned long count );
LONG SetFont( struct RastPort *rp, struct TextFont *textFont );
struct TextFont *OpenFont( struct TextAttr *textAttr );
void CloseFont( struct TextFont *textFont );
ULONG AskSoftStyle( struct RastPort *rp );
ULONG SetSoftStyle( struct RastPort *rp, unsigned long style,
        unsigned long enable );
```

clib/graphics_protos.h Page 2

```
/*------        Gels routines ------*/
void AddBob( struct Bob *bob, struct RastPort *rp );
void AddVSprite( struct VSprite *vSprite, struct RastPort *rp );
void DoCollision( struct RastPort *rp );
void DrawGList( struct RastPort *rp, struct ViewPort *vp );
void InitGels( struct VSprite *head, struct VSprite *tail,
        struct GelsInfo *gelsInfo );
void InitMasks( struct VSprite *vSprite );
void RemIBob( struct Bob *bob, struct RastPort *rp, struct ViewPort *vp );
void RemVSprite( struct VSprite *vSprite );
void SetCollision( unsigned long num,
        void (*routine)(struct VSprite *vSprite, APTR),
        struct GelsInfo *gelsInfo );
void SortGList( struct RastPort *rp );
void AddAnimOb( struct AnimOb *anOb, struct AnimOb **anKey,
        struct RastPort *rp );
void Animate( struct AnimOb **anKey, struct RastPort *rp );
BOOL GetGBuffers( struct AnimOb *anOb, struct RastPort *rp, long flag );
void InitGMasks( struct AnimOb *anOb );
/*------        General graphics routines ------*/
void DrawEllipse( struct RastPort *rp, long xCenter, long yCenter, long a,
        long b );
LONG AreaEllipse( struct RastPort *rp, long xCenter, long yCenter, long a,
        long b );
void LoadRGB4( struct ViewPort *vp, UWORD *colors, long count );
void InitRastPort( struct RastPort *rp );
void InitVPort( struct ViewPort *vp );
void MrgCop( struct View *view );
void MakeVPort( struct View *view, struct ViewPort *vp );
void LoadView( struct View *view );
void WaitBlit( void );
void SetRast( struct RastPort *rp, unsigned long pen );
void Move( struct RastPort *rp, long x, long y );
void Draw( struct RastPort *rp, long x, long y );
LONG AreaMove( struct RastPort *rp, long x, long y );
LONG AreaDraw( struct RastPort *rp, long x, long y );
LONG AreaEnd( struct RastPort *rp );
void WaitTOF( void );
void QBlit( struct bltnode *blit );
void InitArea( struct AreaInfo *areaInfo, APTR vectorBuffer,
        long maxVectors );
void SetRGB4( struct ViewPort *vp, long index, unsigned long red,
        unsigned long green, unsigned long blue );
void QBSBlit( struct bltnode *blit );
void BltClear( PLANEPTR memBlock, unsigned long byteCount,
        unsigned long flags );
void RectFill( struct RastPort *rp, long xMin, long yMin, long xMax,
        long yMax );
void BltPattern( struct RastPort *rp, PLANEPTR mask, long xMin, long yMin,
        long xMax, long yMax, unsigned long maskBPR );
ULONG ReadPixel( struct RastPort *rp, long x, long y );
LONG WritePixel( struct RastPort *rp, long x, long y );
BOOL Flood( struct RastPort *rp, unsigned long mode, long x, long y );
void PolyDraw( struct RastPort *rp, long count, WORD *polyTable );
void SetAPen( struct RastPort *rp, unsigned long pen );
void SetBPen( struct RastPort *rp, unsigned long pen );
void SetDrMd( struct RastPort *rp, unsigned long drawMode );
void InitView( struct View *view );
void CBump( struct UCopList *copList );
void CMove( struct UCopList *copList, APTR destination, long data );
void CWait( struct UCopList *copList, long v, long h );
LONG VBeamPos( void );
void InitBitMap( struct BitMap *bitMap, long depth, long width, long height );
void ScrollRaster( struct RastPort *rp, long dx, long dy, long xMin, long yMin,
        long xMax, long yMax );
void WaitBOVP( struct ViewPort *vp );
```

clib/graphics_protos.h Page 3

```
WORD GetSprite( struct SimpleSprite *sprite, long num );
void FreeSprite( long num );
void ChangeSprite( struct ViewPort *vp, struct SimpleSprite *sprite,
        PLANEPTR newData );
void MoveSprite( struct ViewPort *vp, struct SimpleSprite *sprite, long x,
        long y );
void LockLayerRom( struct Layer *layer );
void UnlockLayerRom( struct Layer *layer );
void SyncSBitMap( struct Layer *layer );
void CopySBitMap( struct Layer *layer );
void OwnBlitter( void );
void DisownBlitter( void );
struct TmpRas *InitTmpRas( struct TmpRas *tmpRas, PLANEPTR buffer,
        long size );
void AskFont( struct RastPort *rp, struct TextAttr *textAttr );
void AddFont( struct TextFont *textFont );
void RemFont( struct TextFont *textFont );
PLANEPTR AllocRaster( unsigned long width, unsigned long height );
void FreeRaster( PLANEPTR p, unsigned long width, unsigned long height );
void AndRectRegion( struct Region *region, struct Rectangle *rectangle );
BOOL OrRectRegion( struct Region *region, struct Rectangle *rectangle );
struct Region *NewRegion( void );
BOOL ClearRectRegion( struct Region *region, struct Rectangle *rectangle );
void ClearRegion( struct Region *region );
void DisposeRegion( struct Region *region );
void FreeVPortCopLists( struct ViewPort *vp );
void FreeCopList( struct CopList *copList );
void ClipBlit( struct RastPort *srcRP, long xSrc, long ySrc,
        struct RastPort *destRP, long xDest, long yDest, long xSize,
        long ySize, unsigned long minterm );
BOOL XorRectRegion( struct Region *region, struct Rectangle *rectangle );
void FreeCprList( struct cprlist *cprList );
struct ColorMap *GetColorMap( long entries );
void FreeColorMap( struct ColorMap *colorMap );
ULONG GetRGB4( struct ColorMap *colorMap, long entry );
void ScrollVPort( struct ViewPort *vp );
struct CopList *UCopperListInit( struct UCopList *uCopList, long n );
void FreeGBuffers( struct AnimOb *anOb, struct RastPort *rp, long flag );
void BltBitMapRastPort( struct BitMap *srcBitMap, long xSrc, long ySrc,
        struct RastPort *destRP, long xDest, long yDest, long xSize,
        long ySize, unsigned long minterm );
BOOL OrRegionRegion( struct Region *srcRegion, struct Region *destRegion );
BOOL XorRegionRegion( struct Region *srcRegion, struct Region *destRegion );
BOOL AndRegionRegion( struct Region *srcRegion, struct Region *destRegion );
void SetRGB4CM( struct ColorMap *colorMap, long index, unsigned long red,
        unsigned long green, unsigned long blue );
void BltMaskBitMapRastPort( struct BitMap *srcBitMap, long xSrc, long ySrc,
        struct RastPort *destRP, long xDest, long yDest, long xSize,
        long ySize, unsigned long minterm, PLANEPTR bltMask );
BOOL AttemptLockLayerRom( struct Layer *layer );
/*--- functions in V36 or higher (distributed as Release 2.0) ---*/
APTR GfxNew( unsigned long gfxNodeType );
void GfxFree( APTR gfxNodePtr );
void GfxAssociate( APTR associateNode, APTR gfxNodePtr );
void BitMapScale( struct BitScaleArgs *bitScaleArgs );
UWORD ScalerDiv( unsigned long factor, unsigned long numerator,
        unsigned long denominator );
WORD TextExtent( struct RastPort *rp, STRPTR string, long count,
        struct TextExtent *textExtent );
ULONG TextFit( struct RastPort *rp, STRPTR string, unsigned long strLen,
        struct TextExtent *textExtent, struct TextExtent *constrainingExtent,
        long strDirection, unsigned long constrainingBitWidth,
        unsigned long constrainingBitHeight );
APTR GfxLookUp( APTR associateNode );
BOOL VideoControl( struct ColorMap *colorMap, struct TagItem *tagarray );
struct MonitorSpec *OpenMonitor( STRPTR monitorName,
```

clib/graphics_protos.h

Page 4

```
        unsigned long displayID );
BOOL CloseMonitor( struct MonitorSpec *monitorSpec );
DisplayInfoHandle FindDisplayInfo( unsigned long displayID );
ULONG NextDisplayInfo( unsigned long displayID );
ULONG GetDisplayInfoData( DisplayInfoHandle handle, UBYTE *buf,
        unsigned long size, unsigned long tagID, unsigned long displayID );
void FontExtent( struct TextFont *font, struct TextExtent *fontExtent );
LONG ReadPixelLine8( struct RastPort *rp, unsigned long xstart,
        unsigned long ystart, unsigned long width, UBYTE *array,
        struct RastPort *tempRP );
LONG WritePixelLine8( struct RastPort *rp, unsigned long xstart,
        unsigned long ystart, unsigned long width, UBYTE *array,
        struct RastPort *tempRP );
LONG ReadPixelArray8( struct RastPort *rp, unsigned long xstart,
        unsigned long ystart, unsigned long xstop, unsigned long ystop,
        UBYTE *array, struct RastPort *temprp );
LONG WritePixelArray8( struct RastPort *rp, unsigned long xstart,
        unsigned long ystart, unsigned long xstop, unsigned long ystop,
        UBYTE *array, struct RastPort *temprp );
LONG GetVPModeID( struct ViewPort *vp );
LONG ModeNotAvailable( unsigned long modeID );
WORD WeighTAMatch( struct TextAttr *reqTextAttr,
        struct TextAttr *targetTextAttr, struct TagItem *targetTags );
void EraseRect( struct RastPort *rp, long xMin, long yMin, long xMax,
        long yMax );
ULONG ExtendFont( struct TextFont *font, struct TagItem *fontTags );
void StripFont( struct TextFont *font );
#endif   /* CLIB_GRAPHICS_PROTOS_H */
```

clib/icon_protos.h

Page 1

```
#ifndef  CLIB_ICON_PROTOS_H
#define  CLIB_ICON_PROTOS_H
/*
**      $Filename: clib/icon_protos.h $
**      $Release: 2.04 $
**      $Revision: 37.1 $
**      $Date: 91/03/01 $
**
**      C prototypes. For use with 32 bit integers only.
**
**      (C) Copyright 1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/
/* "icon.library" */
/*--- functions in V36 or higher (distributed as Release 2.0) ---*/
#ifndef  EXEC_TYPES_H
#include <exec/types.h>
#endif
#ifndef  WORKBENCH_WORKBENCH_H
#include <workbench/workbench.h>
#endif
/*      Use DiskObjects instead of obsolete WBObjects */
LONG GetIcon( UBYTE *name, struct DiskObject *icon,
        struct FreeList *freelist );
BOOL PutIcon( UBYTE *name, struct DiskObject *icon );
void FreeFreeList( struct FreeList *freelist );
BOOL AddFreeList( struct FreeList *freelist, APTR mem, unsigned long size );
struct DiskObject *GetDiskObject( UBYTE *name );
BOOL PutDiskObject( UBYTE *name, struct DiskObject *diskobj );
void FreeDiskObject( struct DiskObject *diskobj );
UBYTE *FindToolType( UBYTE **toolTypeArray, UBYTE *typeName );
BOOL MatchToolValue( UBYTE *typeString, UBYTE *value );
UBYTE *BumpRevision( UBYTE *newname, UBYTE *oldname );
struct DiskObject *GetDefDiskObject( long type );
BOOL PutDefDiskObject( struct DiskObject *diskObject );
struct DiskObject *GetDiskObjectNew( UBYTE *name );
BOOL DeleteDiskObject( UBYTE *name );
#endif   /* CLIB_ICON_PROTOS_H */
```

clib/iffparse_protos.h Page 1

```
#ifndef  CLIB_IFFPARSE_PROTOS_H
#define  CLIB_IFFPARSE_PROTOS_H
/*
**      $Filename: clib/iffparse_protos.h $
**      $Release: 2.04 $
**      $Revision: 33.2 $
**      $Date: 90/12/13 $
**
**      C prototypes. For use with 32 bit integers only.
**
**      (C) Copyright 1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/
/* "iffparse.library" */
#ifndef  EXEC_TYPES_H
#include <exec/types.h>
#endif
/*--- functions in V33 or higher (distributed as Release 1.2) ---*/
/*------ Basic functions ------*/
struct IFFHandle *AllocIFF( void );
LONG OpenIFF( struct IFFHandle *iff, long rwMode );
LONG ParseIFF( struct IFFHandle *iff, long control );
void CloseIFF( struct IFFHandle *iff );
void FreeIFF( struct IFFHandle *iff );
/*------ Read/Write functions ------*/
LONG ReadChunkBytes( struct IFFHandle *iff, APTR buf, long size );
LONG WriteChunkBytes( struct IFFHandle *iff, APTR buf, long size );
LONG ReadChunkRecords( struct IFFHandle *iff, APTR buf, long bytesPerRecord,
        long nRecords );
LONG WriteChunkRecords( struct IFFHandle *iff, APTR buf, long bytesPerRecord,
        long nRecords );
/*------ Context entry/exit ------*/
LONG PushChunk( struct IFFHandle *iff, long type, long id, long size );
LONG PopChunk( struct IFFHandle *iff );
/*------ Low-level handler installation ------*/
LONG EntryHandler( struct IFFHandle *iff, long type, long id, long position,
        struct Hook *handler, APTR object );
LONG ExitHandler( struct IFFHandle *iff, long type, long id, long position,
        struct Hook *handler, APTR object );
/*------ Built-in chunk/property handlers ------*/
LONG PropChunk( struct IFFHandle *iff, long type, long id );
LONG PropChunks( struct IFFHandle *iff, LONG *propArray, long nProps );
LONG StopChunk( struct IFFHandle *iff, long type, long id );
LONG StopChunks( struct IFFHandle *iff, LONG *propArray, long nProps );
LONG CollectionChunk( struct IFFHandle *iff, long type, long id );
LONG CollectionChunks( struct IFFHandle *iff, LONG *propArray, long nProps );
LONG StopOnExit( struct IFFHandle *iff, long type, long id );
/*------ Context utilities ------*/
struct StoredProperty *FindProp( struct IFFHandle *iff, long type, long id );
struct CollectionItem *FindCollection( struct IFFHandle *iff, long type,
        long id );
struct ContextNode *FindPropContext( struct IFFHandle *iff );
struct ContextNode *CurrentChunk( struct IFFHandle *iff );
struct ContextNode *ParentChunk( struct ContextNode *contextNode );
/*------ LocalContextItem support functions ------*/
struct LocalContextItem *AllocLocalItem( long type, long id, long ident,
        long dataSize );
APTR LocalItemData( struct LocalContextItem *localItem );
void SetLocalItemPurge( struct LocalContextItem *localItem,
        struct Hook *purgeHook );
void FreeLocalItem( struct LocalContextItem *localItem );
struct LocalContextItem *FindLocalItem( struct IFFHandle *iff, long type,
        long id, long ident );
LONG StoreLocalItem( struct IFFHandle *iff, struct LocalContextItem *localItem,
        long position );
void StoreItemInContext( struct IFFHandle *iff,
```

clib/iffparse_protos.h Page 2

```
        struct LocalContextItem *localItem,
        struct ContextNode *contextNode );
/*------ IFFHandle initialization ------*/
void InitIFF( struct IFFHandle *iff, long flags, struct Hook *streamHook );
void InitIFFasDOS( struct IFFHandle *iff );
void InitIFFasClip( struct IFFHandle *iff );
/*------ Internal clipboard support ------*/
struct ClipboardHandle *OpenClipboard( long unitNum );
void CloseClipboard( struct ClipboardHandle *clipboard );
/*------ Miscellaneous ------*/
LONG GoodID( long id );
LONG GoodType( long type );
STRPTR IDtoStr( long id, STRPTR buf );
#endif   /* CLIB_IFFPARSE_PROTOS_H */
```

clib/input_protos.h

```
#ifndef  CLIB_INPUT_PROTOS_H
#define  CLIB_INPUT_PROTOS_H
/*
**      $Filename: clib/input_protos.h $
**      $Release: 2.04 $
**      $Revision: 36.2 $
**      $Date: 90/11/07 $
**
**      C prototypes. For use with 32 bit integers only.
**
**      (C) Copyright 1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/
/* "input.device" */
#ifndef  EXEC_TYPES_H
#include <exec/types.h>
#endif
/*--- functions in V36 or higher (distributed as Release 2.0) ---*/
UWORD PeekQualifier( void );
#endif   /* CLIB_INPUT_PROTOS_H */
```

clib/intuition_protos.h

```
#ifndef  CLIB_INTUITION_PROTOS_H
#define  CLIB_INTUITION_PROTOS_H
/*
**      $Filename: clib/intuition_protos.h $
**      $Release: 2.04 $
**      $Revision: 36.4 $
**      $Date: 91/02/22 $
**
**      C prototypes. For use with 32 bit integers only.
**
**      (C) Copyright 1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/
/* "intuition.library" */
#ifndef  INTUITION_INTUITION_H
#include <intuition/intuition.h>
#endif
/* Public functions OpenIntuition() and Intuition() are intentionally */
/* not documented. */
void OpenIntuition( void );
void Intuition( struct InputEvent *iEvent );
UWORD AddGadget( struct Window *window, struct Gadget *gadget,
        unsigned long position );
BOOL ClearDMRequest( struct Window *window );
void ClearMenuStrip( struct Window *window );
void ClearPointer( struct Window *window );
BOOL CloseScreen( struct Screen *screen );
void CloseWindow( struct Window *window );
LONG CloseWorkBench( void );
void CurrentTime( ULONG *seconds, ULONG *micros );
BOOL DisplayAlert( unsigned long alertNumber, UBYTE *string,
        unsigned long height );
void DisplayBeep( struct Screen *screen );
BOOL DoubleClick( unsigned long sSeconds, unsigned long sMicros,
        unsigned long cSeconds, unsigned long cMicros );
void DrawBorder( struct RastPort *rp, struct Border *border, long leftOffset,
        long topOffset );
void DrawImage( struct RastPort *rp, struct Image *image, long leftOffset,
        long topOffset );
void EndRequest( struct Requester *requester, struct Window *window );
struct Preferences *GetDefPrefs( struct Preferences *preferences, long size );
struct Preferences *GetPrefs( struct Preferences *preferences, long size );
void InitRequester( struct Requester *requester );
struct MenuItem *ItemAddress( struct Menu *menuStrip,
        unsigned long menuNumber );
BOOL ModifyIDCMP( struct Window *window, unsigned long flags );
void ModifyProp( struct Gadget *gadget, struct Window *window,
        struct Requester *requester, unsigned long flags,
        unsigned long horizPot, unsigned long vertPot,
        unsigned long horizBody, unsigned long vertBody );
void MoveScreen( struct Screen *screen, long dx, long dy );
void MoveWindow( struct Window *window, long dx, long dy );
void OffGadget( struct Gadget *gadget, struct Window *window,
        struct Requester *requester );
void OffMenu( struct Window *window, unsigned long menuNumber );
void OnGadget( struct Gadget *gadget, struct Window *window,
        struct Requester *requester );
void OnMenu( struct Window *window, unsigned long menuNumber );
struct Screen *OpenScreen( struct NewScreen *newScreen );
struct Window *OpenWindow( struct NewWindow *newWindow );
BOOL OpenWorkBench( void );
void PrintIText( struct RastPort *rp, struct IntuiText *iText, long left,
        long top );
void RefreshGadgets( struct Gadget *gadgets, struct Window *window,
        struct Requester *requester );
UWORD RemoveGadget( struct Window *window, struct Gadget *gadget );
```

clib/intuition_protos.h Page 2

```
/* The official calling sequence for ReportMouse is given below. */
/* Note the register order.  For the complete story, read the ReportMouse */
/* autodoc. */
void ReportMouse( long flag, struct Window *window );
void ReportMouse1( struct Window *window, long flag );
BOOL Request( struct Requester *requester, struct Window *window );
void ScreenToBack( struct Screen *screen );
void ScreenToFront( struct Screen *screen );
BOOL SetDMRequest( struct Window *window, struct Requester *requester );
BOOL SetMenuStrip( struct Window *window, struct Menu *menu );
void SetPointer( struct Window *window, UWORD *pointer, long height,
        long width, long xOffset, long yOffset );
void SetWindowTitles( struct Window *window, UBYTE *windowTitle,
        UBYTE *screenTitle );
void ShowTitle( struct Screen *screen, long showIt );
void SizeWindow( struct Window *window, long dx, long dy );
struct View *ViewAddress( void );
struct ViewPort *ViewPortAddress( struct Window *window );
void WindowToBack( struct Window *window );
void WindowToFront( struct Window *window );
BOOL WindowLimits( struct Window *window, long widthMin, long heightMin,
        unsigned long widthMax, unsigned long heightMax );
/*--- start of next generation of names -------------------------------------*/
struct Preferences *SetPrefs( struct Preferences *preferences, long size,
        long inform );
/*--- start of next next generation of names --------------------------------*/
LONG IntuiTextLength( struct IntuiText *iText );
BOOL WBenchToBack( void );
BOOL WBenchToFront( void );
/*--- start of next next next generation of names ---------------------------*/
BOOL AutoRequest( struct Window *window, struct IntuiText *body,
        struct IntuiText *posText, struct IntuiText *negText,
        unsigned long pFlag, unsigned long nFlag, unsigned long width,
        unsigned long height );
void BeginRefresh( struct Window *window );
struct Window *BuildSysRequest( struct Window *window, struct IntuiText *body,
        struct IntuiText *posText, struct IntuiText *negText,
        unsigned long flags, unsigned long width, unsigned long height );
void EndRefresh( struct Window *window, long complete );
void FreeSysRequest( struct Window *window );
void MakeScreen( struct Screen *screen );
void RemakeDisplay( void );
void RethinkDisplay( void );
/*--- start of next next next next generation of names ----------------------*/
APTR AllocRemember( struct Remember **rememberKey, unsigned long size,
        unsigned long flags );
/* Public function AlohaWorkbench() is intentionally not documented */
void AlohaWorkbench( long wbport );
void FreeRemember( struct Remember **rememberKey, long reallyForget );
/*--- start of 15 Nov 85 names -----------------------*/
ULONG LockIBase( unsigned long dontknow );
void UnlockIBase( unsigned long ibLock );
/*--- functions in V33 or higher (distributed as Release 1.2) ---*/
LONG GetScreenData( APTR buffer, unsigned long size, unsigned long type,
        struct Screen *screen );
void RefreshGList( struct Gadget *gadgets, struct Window *window,
        struct Requester *requester, long numGad );
UWORD AddGList( struct Window *window, struct Gadget *gadget,
        unsigned long position, long numGad, struct Requester *requester );
UWORD RemoveGList( struct Window *remPtr, struct Gadget *gadget,
        long numGad );
LONG ActivateWindow( struct Window *window );
void RefreshWindowFrame( struct Window *window );
BOOL ActivateGadget( struct Gadget *gadgets, struct Window *window,
        struct Requester *requester );
void NewModifyProp( struct Gadget *gadget, struct Window *window,
```

clib/intuition_protos.h Page 3

```
        struct Requester *requester, unsigned long flags,
        unsigned long horizPot, unsigned long vertPot,
        unsigned long horizBody, unsigned long vertBody, long numGad );
/*--- functions in V36 or higher (distributed as Release 2.0) ---*/
LONG QueryOverscan( unsigned long displayID, struct Rectangle *rect,
        long oScanType );
void MoveWindowInFrontOf( struct Window *window,
        struct Window *behindWindow );
void ChangeWindowBox( struct Window *window, long left, long top, long width,
        long height );
struct Hook *SetEditHook( struct Hook *hook );
LONG SetMouseQueue( struct Window *window, unsigned long queueLength );
void ZipWindow( struct Window *window );
/*--- public screens ---*/
struct Screen *LockPubScreen( UBYTE *name );
void UnlockPubScreen( UBYTE *name, struct Screen *screen );
struct List *LockPubScreenList( void );
void UnlockPubScreenList( void );
UBYTE *NextPubScreen( struct Screen *screen, UBYTE *namebuf );
void SetDefaultPubScreen( UBYTE *name );
UWORD SetPubScreenModes( unsigned long modes );
UWORD PubScreenStatus( struct Screen *screen, unsigned long statusFlags );
/**/
struct RastPort *ObtainGIRPort( struct GadgetInfo *gInfo );
void ReleaseGIRPort( struct RastPort *rp );
void GadgetMouse( struct Gadget *gadget, struct GadgetInfo *gInfo,
        WORD *mousePoint );
/* SetIPrefs is system private and not to be used by applications */
void GetDefaultPubScreen( UBYTE *nameBuffer );
LONG EasyRequestArgs( struct Window *window, struct EasyStruct *easyStruct,
        ULONG *idcmpPtr, APTR args );
LONG EasyRequest( struct Window *window, struct EasyStruct *easyStruct,
        ULONG *idcmpPtr, APTR arg1, ... );
struct Window *BuildEasyRequestArgs( struct Window *window,
        struct EasyStruct *easyStruct, unsigned long idcmp, APTR args );
struct Window *BuildEasyRequest( struct Window *window,
        struct EasyStruct *easyStruct, unsigned long idcmp, APTR arg1, ... );
LONG SysReqHandler( struct Window *window, ULONG *idcmpPtr, long waitInput );
struct Window *OpenWindowTagList( struct NewWindow *newWindow,
        struct TagItem *tagList );
struct Window *OpenWindowTags( struct NewWindow *newWindow,
        unsigned long tag1Type, ... );
struct Screen *OpenScreenTagList( struct NewScreen *newScreen,
        struct TagItem *tagList );
struct Screen *OpenScreenTags( struct NewScreen *newScreen,
        unsigned long tag1Type, ... );
/**/
/*      new Image functions */
void DrawImageState( struct RastPort *rp, struct Image *image, long leftOffset,
        long topOffset, unsigned long state, struct DrawInfo *drawInfo );
BOOL PointInImage( unsigned long point, struct Image *image );
void EraseImage( struct RastPort *rp, struct Image *image, long leftOffset,
        long topOffset );
/**/
APTR NewObjectA( struct IClass *class, UBYTE *classID,
        struct TagItem *tagList );
APTR NewObject( struct IClass *class, UBYTE *classID, unsigned long tag1,
        ... );
/**/
void DisposeObject( APTR object );
ULONG SetAttrsA( APTR object, struct TagItem *tagList );
ULONG SetAttrs( APTR object, unsigned long tag1, ... );
/**/
ULONG GetAttr( unsigned long attrID, APTR object, ULONG *storagePtr );
/**/
/*      special set attribute call for gadgets */
```

clib/intuition_protos.h

```
ULONG SetGadgetAttrsA( struct Gadget *gadget, struct Window *window,
        struct Requester *requester, struct TagItem *tagList );
ULONG SetGadgetAttrs( struct Gadget *gadget, struct Window *window,
        struct Requester *requester, unsigned long tag1, ... );
/**/
/*      for class implementors only */
APTR NextObject( APTR objectPtrPtr );
struct IClass *MakeClass( UBYTE *classID, UBYTE *superClassID,
        struct IClass *superClassPtr, unsigned long instanceSize,
        unsigned long flags );
void AddClass( struct IClass *class );
/**/
/**/
struct DrawInfo *GetScreenDrawInfo( struct Screen *screen );
void FreeScreenDrawInfo( struct Screen *screen, struct DrawInfo *drawInfo );
/**/
BOOL ResetMenuStrip( struct Window *window, struct Menu *menu );
void RemoveClass( struct IClass *classPtr );
BOOL FreeClass( struct IClass *classPtr );
#endif   /* CLIB_INTUITION_PROTOS_H */
```

clib/keymap_protos.h

```
#ifndef  CLIB_KEYMAP_PROTOS_H
#define  CLIB_KEYMAP_PROTOS_H
/*
**      $Filename: clib/keymap_protos.h $
**      $Release: 2.04 $
**      $Revision: 36.4 $
**      $Date: 90/07/19 $
**
**      C prototypes. For use with 32 bit integers only.
**
**      (C) Copyright 1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/
/* "keymap.library" */
#ifndef  DEVICES_INPUTEVENT_H
#include <devices/inputevent.h>
#endif
#ifndef  DEVICES_KEYMAP_H
#include <devices/keymap.h>
#endif
/*--- functions in V36 or higher (distributed as Release 2.0) ---*/
void SetKeyMapDefault( struct KeyMap *keyMap );
struct KeyMap *AskKeyMapDefault( void );
WORD MapRawKey( struct InputEvent *event, STRPTR buffer, long length,
        struct KeyMap *keyMap );
LONG MapANSI( STRPTR string, long count, STRPTR buffer, long length,
        struct KeyMap *keyMap );
#endif   /* CLIB_KEYMAP_PROTOS_H */
```

clib/layers_protos.h

Page 1

```
#ifndef  CLIB_LAYERS_PROTOS_H
#define  CLIB_LAYERS_PROTOS_H
/*
**      $Filename: clib/layers_protos.h $
**      $Release: 2.04 $
**      $Revision: 1.6 $
**      $Date: 90/12/09 $
**
**      C prototypes. For use with 32 bit integers only.
**
**      (C) Copyright 1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/
/* "layers.library" */
#ifndef  EXEC_TYPES_H
#include <exec/types.h>
#endif
#ifndef  GRAPHICS_LAYERS_H
#include <graphics/layers.h>
#endif
#ifndef  GRAPHICS_CLIP_H
#include <graphics/clip.h>
#endif
#ifndef  GRAPHICS_RASTPORT_H
#include <graphics/rastport.h>
#endif
#ifndef  GRAPHICS_REGIONS_H
#include <graphics/regions.h>
#endif
void InitLayers( struct Layer_Info *li );
struct Layer *CreateUpfrontLayer( struct Layer_Info *li, struct BitMap *bm,
        long x0, long y0, long x1, long y1, long flags, struct BitMap *bm2 );
struct Layer *CreateBehindLayer( struct Layer_Info *li, struct BitMap *bm,
        long x0, long y0, long x1, long y1, long flags, struct BitMap *bm2 );
LONG UpfrontLayer( long dummy, struct Layer *layer );
LONG BehindLayer( long dummy, struct Layer *layer );
LONG MoveLayer( long dummy, struct Layer *layer, long dx, long dy );
LONG SizeLayer( long dummy, struct Layer *layer, long dx, long dy );
void ScrollLayer( long dummy, struct Layer *layer, long dx, long dy );
LONG BeginUpdate( struct Layer *l );
void EndUpdate( struct Layer *layer, unsigned long flag );
LONG DeleteLayer( long dummy, struct Layer *layer );
void LockLayer( long dummy, struct Layer *layer );
void UnlockLayer( struct Layer *layer );
void LockLayers( struct Layer_Info *li );
void UnlockLayers( struct Layer_Info *li );
void LockLayerInfo( struct Layer_Info *li );
void SwapBitsRastPortClipRect( struct RastPort *rp, struct ClipRect *cr );
struct Layer *WhichLayer( struct Layer_Info *li, long x, long y );
void UnlockLayerInfo( struct Layer_Info *li );
struct Layer_Info *NewLayerInfo( void );
void DisposeLayerInfo( struct Layer_Info *li );
LONG FattenLayerInfo( struct Layer_Info *li );
void ThinLayerInfo( struct Layer_Info *li );
LONG MoveLayerInFrontOf( struct Layer *layer_to_move,
        struct Layer *other_layer );
struct Region *InstallClipRegion( struct Layer *layer,struct Region *region );
LONG MoveSizeLayer( struct Layer *layer, long dx, long dy, long dw, long dh );
struct Layer *CreateUpfrontHookLayer( struct Layer_Info *li, struct BitMap *bm,
        long x0, long y0, long x1, long y1, long flags, struct Hook *hook,
        struct BitMap *bm2 );
struct Layer *CreateBehindHookLayer( struct Layer_Info *li, struct BitMap *bm,
        long x0, long y0, long x1, long y1, long flags, struct Hook *hook,
        struct BitMap *bm2 );
struct Hook *InstallLayerHook( struct Layer *layer, struct Hook *hook );
#endif   /* CLIB_LAYERS_PROTOS_H */
```

clib/macros.h

Page 1

```
#ifndef CLIB_MACROS_H
#define CLIB_MACROS_H
/*
**      $Filename: clib/macros.h $
**      $Release: 2.04 $
**      $Revision: 36.0 $
**      $Date: 90/11/30 $
**
**      C prototypes
**
**      (C) Copyright 1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#define MAX(a,b)     ((a)>(b)?(a):(b))
#define MIN(a,b)     ((a)<(b)?(a):(b))
#define ABS(x)       ((x<0)?(-(x)):(x))

#endif  /* CLIB_MACROS_H */
```

clib/mathffp_protos.h

Page 1

```
#ifndef  CLIB_MATHFFP_PROTOS_H
#define  CLIB_MATHFFP_PROTOS_H
/*
**      $Filename: clib/mathffp_protos.h $
**      $Release: 2.04 $
**      $Revision: 1.4 $
**      $Date: 90/05/03 $
**
**      C prototypes. For use with 32 bit integers only.
**
**      (C) Copyright 1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/
/* "mathffp.library" */
#ifndef  EXEC_TYPES_H
#include <exec/types.h>
#endif
LONG SPFix( FLOAT parm );
FLOAT SPFlt( long integer );
LONG SPCmp( FLOAT leftParm, FLOAT rightParm );
LONG SPTst( FLOAT parm );
FLOAT SPAbs( FLOAT parm );
FLOAT SPNeg( FLOAT parm );
FLOAT SPAdd( FLOAT leftParm, FLOAT rightParm );
FLOAT SPSub( FLOAT leftParm, FLOAT rightParm );
FLOAT SPMul( FLOAT leftParm, FLOAT rightParm );
FLOAT SPDiv( FLOAT leftParm, FLOAT rightParm );
/*--- functions in V33 or higher (distributed as Release 1.2) ---*/
FLOAT SPFloor( FLOAT parm );
FLOAT SPCeil( FLOAT parm );
#endif   /* CLIB_MATHFFP_PROTOS_H */
```

clib/mathieeedoubbas_protos.h

Page 1

```
#ifndef  CLIB_MATHIEEEDOUBBAS_PROTOS_H
#define  CLIB_MATHIEEEDOUBBAS_PROTOS_H
/*
**      $Filename: clib/mathieeedoubbas_protos.h $
**      $Release: 2.04 $
**      $Revision: 1.3 $
**      $Date: 90/11/07 $
**
**      C prototypes. For use with 32 bit integers only.
**
**      (C) Copyright 1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/
/* "mathieeedoubbas.library" */
#ifndef  EXEC_TYPES_H
#include <exec/types.h>
#endif
LONG IEEEDPFix( DOUBLE parm );
DOUBLE IEEEDPFlt( long integer );
LONG IEEEDPCmp( DOUBLE leftParm, DOUBLE rightParm );
LONG IEEEDPTst( DOUBLE parm );
DOUBLE IEEEDPAbs( DOUBLE parm );
DOUBLE IEEEDPNeg( DOUBLE parm );
DOUBLE IEEEDPAdd( DOUBLE leftParm, DOUBLE rightParm );
DOUBLE IEEEDPSub( DOUBLE leftParm, DOUBLE rightParm );
DOUBLE IEEEDPMul( DOUBLE factor1, DOUBLE factor2 );
DOUBLE IEEEDPDiv( DOUBLE dividend, DOUBLE divisor );
/*--- functions in V33 or higher (distributed as Release 1.2) ---*/
DOUBLE IEEEDPFloor( DOUBLE parm );
DOUBLE IEEEDPCeil( DOUBLE parm );
#endif   /* CLIB_MATHIEEEDOUBBAS_PROTOS_H */
```

clib/mathieeedoubtrans_protos.h

```
#ifndef  CLIB_MATHIEEEDOUBTRANS_PROTOS_H
#define  CLIB_MATHIEEEDOUBTRANS_PROTOS_H
/*
**      $Filename: clib/mathieeedoubtrans_protos.h $
**      $Release: 2.04 $
**      $Revision: 1.3 $
**      $Date: 90/11/07 $
**
**      C prototypes. For use with 32 bit integers only.
**
**      (C) Copyright 1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/
/* "mathieeedoubtrans.library" */
#ifndef  EXEC_TYPES_H
#include <exec/types.h>
#endif
DOUBLE IEEEDPAtan( DOUBLE parm );
DOUBLE IEEEDPSin( DOUBLE parm );
DOUBLE IEEEDPCos( DOUBLE parm );
DOUBLE IEEEDPTan( DOUBLE parm );
DOUBLE IEEEDPSincos( DOUBLE *pf2, DOUBLE parm );
DOUBLE IEEEDPSinh( DOUBLE parm );
DOUBLE IEEEDPCosh( DOUBLE parm );
DOUBLE IEEEDPTanh( DOUBLE parm );
DOUBLE IEEEDPExp( DOUBLE parm );
DOUBLE IEEEDPLog( DOUBLE parm );
DOUBLE IEEEDPPow( DOUBLE exp, DOUBLE arg );
DOUBLE IEEEDPSqrt( DOUBLE parm );
FLOAT IEEEDPTieee( DOUBLE parm );
DOUBLE IEEEDPFieee( FLOAT single );
DOUBLE IEEEDPAsin( DOUBLE parm );
DOUBLE IEEEDPAcos( DOUBLE parm );
DOUBLE IEEEDPLog10( DOUBLE parm );
#endif   /* CLIB_MATHIEEEDOUBTRANS_PROTOS_H */
```

clib/mathieeesingbas_protos.h

```
#ifndef  CLIB_MATHIEEESINGBAS_PROTOS_H
#define  CLIB_MATHIEEESINGBAS_PROTOS_H
/*
**      $Filename: clib/mathieeesingbas_protos.h $
**      $Release: 2.04 $
**      $Revision: 1.3 $
**      $Date: 91/01/13 $
**
**      C prototypes. For use with 32 bit integers only.
**
**      (C) Copyright 1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/
/* "mathieeesingbas.library" */
#ifndef  EXEC_TYPES_H
#include <exec/types.h>
#endif
LONG IEEESPFix( FLOAT parm );
FLOAT IEEESPFlt( long integer );
LONG IEEESPCmp( FLOAT leftParm, FLOAT rightParm );
LONG IEEESPTst( FLOAT parm );
FLOAT IEEESPAbs( FLOAT parm );
FLOAT IEEESPNeg( FLOAT parm );
FLOAT IEEESPAdd( FLOAT leftParm, FLOAT rightParm );
FLOAT IEEESPSub( FLOAT leftParm, FLOAT rightParm );
FLOAT IEEESPMul( FLOAT leftParm, FLOAT rightParm );
FLOAT IEEESPDiv( FLOAT dividend, FLOAT divisor );
FLOAT IEEESPFloor( FLOAT parm );
FLOAT IEEESPCeil( FLOAT parm );
#endif   /* CLIB_MATHIEEESINGBAS_PROTOS_H */
```

clib/mathieeesingtrans_protos.h

```
#ifndef  CLIB_MATHIEEESINGTRANS_PROTOS_H
#define  CLIB_MATHIEEESINGTRANS_PROTOS_H
/*
**      $Filename: clib/mathieeesingtrans_protos.h $
**      $Release: 2.04 $
**      $Revision: 1.3 $
**      $Date: 90/11/07 $
**
**      C prototypes. For use with 32 bit integers only.
**
**      (C) Copyright 1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/
/* "mathieeesingtrans.library" */
#ifndef  EXEC_TYPES_H
#include <exec/types.h>
#endif
FLOAT IEEESPAtan( FLOAT parm );
FLOAT IEEESPSin( FLOAT parm );
FLOAT IEEESPCos( FLOAT parm );
FLOAT IEEESPTan( FLOAT parm );
FLOAT IEEESPSincos( FLOAT *cosptr, FLOAT parm );
FLOAT IEEESPSinh( FLOAT parm );
FLOAT IEEESPCosh( FLOAT parm );
FLOAT IEEESPTanh( FLOAT parm );
FLOAT IEEESPExp( FLOAT parm );
FLOAT IEEESPLog( FLOAT parm );
FLOAT IEEESPPow( FLOAT exp, FLOAT arg );
FLOAT IEEESPSqrt( FLOAT parm );
FLOAT IEEESPTieee( FLOAT parm );
FLOAT IEEESPFieee( FLOAT parm );
FLOAT IEEESPAsin( FLOAT parm );
FLOAT IEEESPAcos( FLOAT parm );
FLOAT IEEESPLog10( FLOAT parm );
#endif   /* CLIB_MATHIEEESINGTRANS_PROTOS_H */
```

clib/mathtrans_protos.h

```
#ifndef  CLIB_MATHTRANS_PROTOS_H
#define  CLIB_MATHTRANS_PROTOS_H
/*
**      $Filename: clib/mathtrans_protos.h $
**      $Release: 2.04 $
**      $Revision: 1.2 $
**      $Date: 90/11/07 $
**
**      C prototypes. For use with 32 bit integers only.
**
**      (C) Copyright 1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/
/* "mathtrans.library" */
#ifndef  EXEC_TYPES_H
#include <exec/types.h>
#endif
FLOAT SPAtan( FLOAT parm );
FLOAT SPSin( FLOAT parm );
FLOAT SPCos( FLOAT parm );
FLOAT SPTan( FLOAT parm );
FLOAT SPSincos( FLOAT *cosResult, FLOAT parm );
FLOAT SPSinh( FLOAT parm );
FLOAT SPCosh( FLOAT parm );
FLOAT SPTanh( FLOAT parm );
FLOAT SPExp( FLOAT parm );
FLOAT SPLog( FLOAT parm );
FLOAT SPPow( FLOAT power, FLOAT arg );
FLOAT SPSqrt( FLOAT parm );
FLOAT SPTieee( FLOAT parm );
FLOAT SPFieee( FLOAT parm );
/*--- functions in V31 or higher (distributed as Release 1.1) ---*/
FLOAT SPAsin( FLOAT parm );
FLOAT SPAcos( FLOAT parm );
FLOAT SPLog10( FLOAT parm );
#endif   /* CLIB_MATHTRANS_PROTOS_H */
```

clib/misc_protos.h

```
#ifndef  CLIB_MISC_PROTOS_H
#define  CLIB_MISC_PROTOS_H
/*
**      $Filename: clib/misc_protos.h $
**      $Release: 2.04 $
**      $Revision: 36.2 $
**      $Date: 90/11/07 $
**
**      C prototypes. For use with 32 bit integers only.
**
**      (C) Copyright 1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/
#ifndef  EXEC_TYPES_H
#include <exec/types.h>
#endif
UBYTE *AllocMiscResource( unsigned long unitNum, UBYTE *name );
void FreeMiscResource( unsigned long unitNum );
#endif   /* CLIB_MISC_PROTOS_H */
```

clib/potgo_protos.h

```
#ifndef  CLIB_POTGO_PROTOS_H
#define  CLIB_POTGO_PROTOS_H
/*
**      $Filename: clib/potgo_protos.h $
**      $Release: 2.04 $
**      $Revision: 36.3 $
**      $Date: 90/11/07 $
**
**      C prototypes. For use with 32 bit integers only.
**
**      (C) Copyright 1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/
/* "potgo.resource" */
#ifndef  EXEC_TYPES_H
#include <exec/types.h>
#endif
UWORD AllocPotBits( unsigned long bits );
void FreePotBits( unsigned long bits );
void WritePotgo( unsigned long word, unsigned long mask );
#endif   /* CLIB_POTGO_PROTOS_H */
```

clib/ramdrive_protos.h

```
#ifndef  CLIB_RAMDRIVE_PROTOS_H
#define  CLIB_RAMDRIVE_PROTOS_H
/*
**      $Filename: clib/ramdrive_protos.h $
**      $Release: 2.04 $
**      $Revision: 36.3 $
**      $Date: 90/11/07 $
**
**      C prototypes. For use with 32 bit integers only.
**
**      (C) Copyright 1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/
/* "ramdrive.device" */
#ifndef  EXEC_TYPES_H
#include <exec/types.h>
#endif
/*--- functions in V34 or higher (distributed as Release 1.3) ---*/
STRPTR KillRAD0( void );
/*--- functions in V36 or higher (distributed as Release 2.0) ---*/
STRPTR KillRAD( unsigned long unit );
#endif   /* CLIB_RAMDRIVE_PROTOS_H */
```

clib/rexxsyslib_protos.h

```
#ifndef  CLIB_REXXSYSLIB_PROTOS_H
#define  CLIB_REXXSYSLIB_PROTOS_H
/*
**      $Filename: clib/rexxsyslib_protos.h $
**      $Release: 2.04 $
**      $Revision: 36.3 $
**      $Date: 91/02/19 $
**
**      C prototypes. For use with 32 bit integers only.
**
**      (C) Copyright 1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/
/* "rexxsyslib.library" */
/*--- functions in V33 or higher (distributed as Release 1.2) ---*/
#ifndef  EXEC_TYPES_H
#include <exec/types.h>
#endif
#ifndef  REXX_RXSLIB_H
#include <rexx/rxslib.h>
#endif
#ifndef  REXX_REXXIO_H
#include <rexx/rexxio.h>
#endif
/**/
/**/
UBYTE *CreateArgstring( UBYTE *string, unsigned long length );
void DeleteArgstring( UBYTE *argstring );
ULONG LengthArgstring( UBYTE *argstring );
struct RexxMsg *CreateRexxMsg( struct MsgPort *port, UBYTE *extension,
        UBYTE *host );
void DeleteRexxMsg( struct RexxMsg *packet );
void ClearRexxMsg( struct RexxMsg *msgptr, unsigned long count );
BOOL FillRexxMsg( struct RexxMsg *msgptr, unsigned long count,
        unsigned long mask );
BOOL IsRexxMsg( struct RexxMsg *msgptr );
/**/
/**/
void LockRexxBase( unsigned long resource );
void UnlockRexxBase( unsigned long resource );
/**/
#endif   /* CLIB_REXXSYSLIB_PROTOS_H */
```

clib/timer_protos.h

```
#ifndef  CLIB_TIMER_PROTOS_H
#define  CLIB_TIMER_PROTOS_H
/*
**      $Filename: clib/timer_protos.h $
**      $Release: 2.04 $
**      $Revision: 1.6 $
**      $Date: 91/01/25 $
**
**      C prototypes. For use with 32 bit integers only.
**
**      (C) Copyright 1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/
/* "Timer.Device" */
#ifndef  DEVICES_TIMER_H
#include <devices/timer.h>
#endif
void AddTime( struct timeval *dest, struct timeval *src );
void SubTime( struct timeval *dest, struct timeval *src );
LONG CmpTime( struct timeval *dest, struct timeval *src );
ULONG ReadEClock( struct EClockVal *dest );
void GetSysTime( struct timeval *dest );
#endif   /* CLIB_TIMER_PROTOS_H */
```

clib/translator_protos.h

```
#ifndef  CLIB_TRANSLATOR_PROTOS_H
#define  CLIB_TRANSLATOR_PROTOS_H
/*
**      $Filename: clib/translator_protos.h $
**      $Release: 2.04 $
**      $Revision: 36.1 $
**      $Date: 90/11/07 $
**
**      C prototypes. For use with 32 bit integers only.
**
**      (C) Copyright 1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/
/* "translator.library" */
#ifndef  EXEC_TYPES_H
#include <exec/types.h>
#endif
LONG Translate( STRPTR inputString, long inputLength, STRPTR outputBuffer,
        long bufferSize );
#endif   /* CLIB_TRANSLATOR_PROTOS_H */
```

clib/utility_protos.h

```
#ifndef  CLIB_UTILITY_PROTOS_H
#define  CLIB_UTILITY_PROTOS_H
/*
**      $Filename: clib/utility_protos.h $
**      $Release: 2.04 $
**      $Revision: 36.6 $
**      $Date: 91/02/13 $
**
**      C prototypes. For use with 32 bit integers only.
**
**      (C) Copyright 1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/
/* "utility.library" */
#ifndef  UTILITY_TAGITEM_H
#include <utility/tagitem.h>
#endif
#ifndef  UTILITY_DATE_H
#include <utility/date.h>
#endif
#ifndef  UTILITY_HOOKS_H
#include <utility/hooks.h>
#endif
/* *** TagItem FUNCTIONS *** */
struct TagItem *FindTagItem( Tag tagVal, struct TagItem *tagList );
ULONG GetTagData( Tag tagVal, unsigned long defaultVal,
        struct TagItem *tagList );
ULONG PackBoolTags( unsigned long initialFlags, struct TagItem *tagList,
        struct TagItem *boolMap );
struct TagItem *NextTagItem( struct TagItem **tagListPtr );
void FilterTagChanges( struct TagItem *newTagList, struct TagItem *oldTagList,
        long apply );
void MapTags( struct TagItem *tagList, struct TagItem *mapList,
        long includeMiss );
struct TagItem *AllocateTagItems( unsigned long numItems );
struct TagItem *CloneTagItems( struct TagItem *tagList );
void FreeTagItems( struct TagItem *tagList );
void RefreshTagItemClones( struct TagItem *cloneList,
        struct TagItem *origList );
BOOL TagInArray( Tag tagVal, Tag *tagArray );
LONG FilterTagItems( struct TagItem *tagList, Tag *filterArray, long logic );
/**/
/* *** HOOK FUNCTIONS *** * */
ULONG CallHookPkt( struct Hook *hook, APTR object, APTR paramPacket );
/**/
/* *** DATE FUNCTIONS *** * */
void Amiga2Date( unsigned long amigaTime, struct ClockData *date );
ULONG Date2Amiga( struct ClockData *date );
ULONG CheckDate( struct ClockData *date );
/**/
/* *** 32 BIT MATH FUNCTIONS *** * */
LONG SMult32( long factor1, long factor2 );
ULONG UMult32( unsigned long factor1, unsigned long factor2 );
/* NOTE: Quotient:Remainder returned in d0:d1 */
LONG SDivMod32( long dividend, long divisor );
ULONG UDivMod32( unsigned long dividend, unsigned long divisor );
/**/
/* *** International string routines *** */
LONG Stricmp( UBYTE *string1, UBYTE *string2 );
LONG Strnicmp( UBYTE *string1, UBYTE *string2, long length );
UBYTE ToUpper( unsigned long character );
UBYTE ToLower( unsigned long character );
#endif   /* CLIB_UTILITY_PROTOS_H */
```

clib/wb_protos.h

```
#ifndef  CLIB_WB_PROTOS_H
#define  CLIB_WB_PROTOS_H
/*
**      $Filename: clib/wb_protos.h $
**      $Release: 2.04 $
**      $Revision: 37.2 $
**      $Date: 91/03/01 $
**
**      C prototypes. For use with 32 bit integers only.
**
**      (C) Copyright 1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/
/* "workbench.library" */
/*--- functions in V36 or higher (distributed as Release 2.0) ---*/
#ifndef  EXEC_TYPES_H
#include <exec/types.h>
#endif
#ifndef  WORKBENCH_WORKBENCH_H
#include <workbench/workbench.h>
#endif
#ifndef  INTUITION_INTUITION_H
#include <intuition/intuition.h>
#endif
#ifndef  UTILITY_TAGITEM_H
#include <utility/tagitem.h>
#endif
/**/
/**/
/**/
/**/
/**/
struct AppWindow *AddAppWindowA( unsigned long id, unsigned long userdata,
        struct Window *window, struct MsgPort *msgport,
        struct TagItem *taglist );
struct AppWindow *AddAppWindow( unsigned long id, unsigned long userdata,
        struct Window *window, struct MsgPort *msgport, Tag tag1, ... );
/**/
BOOL RemoveAppWindow( struct AppWindow *appWindow );
/**/
struct AppIcon *AddAppIconA( unsigned long id, unsigned long userdata,
        UBYTE *text, struct MsgPort *msgport, struct FileLock *lock,
        struct DiskObject *diskobj, struct TagItem *taglist );
struct AppIcon *AddAppIcon( unsigned long id, unsigned long userdata,
        UBYTE *text, struct MsgPort *msgport, struct FileLock *lock,
        struct DiskObject *diskobj, Tag tag1, ... );
/**/
BOOL RemoveAppIcon( struct AppIcon *appIcon );
/**/
struct AppMenuItem *AddAppMenuItemA( unsigned long id, unsigned long userdata,
        UBYTE *text, struct MsgPort *msgport, struct TagItem *taglist );
struct AppMenuItem *AddAppMenuItem( unsigned long id, unsigned long userdata,
        UBYTE *text, struct MsgPort *msgport, Tag tag1, ... );
/**/
BOOL RemoveAppMenuItem( struct AppMenuItem *appMenuItem );
/**/
/**/
#endif   /* CLIB_WB_PROTOS_H */
```

devices/audio.h

Page 1

```
#ifndef DEVICES_AUDIO_H
#define DEVICES_AUDIO_H
/*
**      $Filename: devices/audio.h $
**      $Release: 2.04 $
**      $Revision: 36.3 $
**      $Date: 90/08/29 $
**
**      audio.device include file
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_IO_H
#include "exec/io.h"
#endif

#define AUDIONAME               "audio.device"

#define ADHARD_CHANNELS         4

#define ADALLOC_MINPREC         -128
#define ADALLOC_MAXPREC         127

#define ADCMD_FREE              (CMD_NONSTD+0)
#define ADCMD_SETPREC           (CMD_NONSTD+1)
#define ADCMD_FINISH            (CMD_NONSTD+2)
#define ADCMD_PERVOL            (CMD_NONSTD+3)
#define ADCMD_LOCK              (CMD_NONSTD+4)
#define ADCMD_WAITCYCLE         (CMD_NONSTD+5)
#define ADCMD_ALLOCATE          32

#define ADIOB_PERVOL            4
#define ADIOF_PERVOL            (1<<4)
#define ADIOB_SYNCCYCLE         5
#define ADIOF_SYNCCYCLE         (1<<5)
#define ADIOB_NOWAIT            6
#define ADIOF_NOWAIT            (1<<6)
#define ADIOB_WRITEMESSAGE      7
#define ADIOF_WRITEMESSAGE      (1<<7)

#define ADIOERR_NOALLOCATION    -10
#define ADIOERR_ALLOCFAILED     -11
#define ADIOERR_CHANNELSTOLEN   -12

struct IOAudio {
    struct IORequest ioa_Request;
    WORD ioa_AllocKey;
    UBYTE *ioa_Data;
    ULONG ioa_Length;
    UWORD ioa_Period;
    UWORD ioa_Volume;
    UWORD ioa_Cycles;
    struct Message ioa_WriteMsg;
};

#endif  /* DEVICES_AUDIO_H */
```

devices/bootblock.h

Page 1

```
#ifndef DEVICES_BOOTBLOCK_H
#define DEVICES_BOOTBLOCK_H
/*
**      $Filename: devices/bootblock.h $
**      $Release: 2.04 $
**      $Revision: 36.6 $
**      $Date: 90/11/05 $
**
**      floppy BootBlock definition
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_TYPES_H
#include        "exec/types.h"
#endif

struct BootBlock {
        UBYTE   bb_id[4];               /* 4 character identifier */
        LONG    bb_chksum;              /* boot block checksum (balance) */
        LONG    bb_dosblock;            /* reserved for DOS patch */
};

#define         BOOTSECTS       2       /* 1K bootstrap */

#define BBID_DOS        { 'D', 'O', 'S', '\0' }
#define BBID_KICK       { 'K', 'I', 'C', 'K' }

#define BBNAME_DOS      0x444F5300      /* 'DOS\0' */
#define BBNAME_KICK     0x4B49434B      /* 'KICK' */

#endif  /* DEVICES_BOOTBLOCK_H */
```

```
#ifndef     DEVICES_CLIPBOARD_H
#define     DEVICES_CLIPBOARD_H
/*
**      $Filename: devices/clipboard.h $
**      $Release: 2.04 $
**      $Revision: 36.5 $
**      $Date: 90/11/02 $
**
**      clipboard.device structure definitions
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_TYPES_H
#include        "exec/types.h"
#endif
#ifndef EXEC_NODES_H
#include        "exec/nodes.h"
#endif
#ifndef EXEC_LISTS_H
#include        "exec/lists.h"
#endif
#ifndef EXEC_PORTS_H
#include        "exec/ports.h"
#endif

#define CBD_POST                (CMD_NONSTD+0)
#define CBD_CURRENTREADID       (CMD_NONSTD+1)
#define CBD_CURRENTWRITEID      (CMD_NONSTD+2)
#define CBD_CHANGEHOOK          (CMD_NONSTD+3)

#define CBERR_OBSOLETEID        1


struct ClipboardUnitPartial {
    struct  Node cu_Node;       /* list of units */
    ULONG   cu_UnitNum;         /* unit number for this unit */
    /* the remaining unit data is private to the device */
};


struct IOClipReq {
    struct Message io_Message;
    struct Device *io_Device;   /* device node pointer  */
    struct ClipboardUnitPartial *io_Unit; /* unit node pointer */
    UWORD   io_Command;         /* device command */
    UBYTE   io_Flags;           /* including QUICK and SATISFY */
    BYTE    io_Error;           /* error or warning num */
    ULONG   io_Actual;          /* number of bytes transferred */
    ULONG   io_Length;          /* number of bytes requested */
    STRPTR  io_Data;            /* either clip stream or post port */
    ULONG   io_Offset;          /* offset in clip stream */
    LONG    io_ClipID;          /* ordinal clip identifier */
};

#define PRIMARY_CLIP    0       /* primary clip unit */

struct SatisfyMsg {
    struct Message sm_Msg;      /* the length will be 6 */
    UWORD   sm_Unit;            /* which clip unit this is */
    LONG    sm_ClipID;          /* the clip identifier of the post */
};

struct ClipHookMsg {
    ULONG   chm_Type;           /* zero for this structure format */
```

```
    LONG    chm_ChangeCmd;      /* command that caused this hook invocation: */
                                /*   either CMD_UPDATE or CBD_POST */
    LONG    chm_ClipID;         /* the clip identifier of the new data */
};

#endif  /* DEVICES_CLIPBOARD_H */
```

devices/console.h Page 1

```
#ifndef DEVICES_CONSOLE_H
#define DEVICES_CONSOLE_H
/*
**      $Filename: devices/console.h $
**      $Release: 2.04 $
**      $Revision: 36.11 $
**      $Date: 90/11/07 $
**
**      Console device command definitions
**
**      (C) Copyright 1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_TYPES_H
#include        "exec/types.h"
#endif

#ifndef EXEC_IO_H
#include        "exec/io.h"
#endif  /* EXEC_IO_H */

/****** Console commands ******/
#define CD_ASKKEYMAP            (CMD_NONSTD+0)
#define CD_SETKEYMAP            (CMD_NONSTD+1)
#define CD_ASKDEFAULTKEYMAP     (CMD_NONSTD+2)
#define CD_SETDEFAULTKEYMAP     (CMD_NONSTD+3)

/****** SGR parameters ******/

#define SGR_PRIMARY     0
#define SGR_BOLD        1
#define SGR_ITALIC      3
#define SGR_UNDERSCORE  4
#define SGR_NEGATIVE    7

#define SGR_NORMAL      22      /* default foreground color, not bold */
#define SGR_NOTITALIC   23
#define SGR_NOTUNDERSCORE 24
#define SGR_POSITIVE    27

/* these names refer to the ANSI standard, not the implementation */
#define SGR_BLACK       30
#define SGR_RED         31
#define SGR_GREEN       32
#define SGR_YELLOW      33
#define SGR_BLUE        34
#define SGR_MAGENTA     35
#define SGR_CYAN        36
#define SGR_WHITE       37
#define SGR_DEFAULT     39

#define SGR_BLACKBG     40
#define SGR_REDBG       41
#define SGR_GREENBG     42
#define SGR_YELLOWBG    43
#define SGR_BLUEBG      44
#define SGR_MAGENTABG   45
#define SGR_CYANBG      46
#define SGR_WHITEBG     47
#define SGR_DEFAULTBG   49

/* these names refer to the implementation, they are the preferred */
/* names for use with the Amiga console device. */
#define SGR_CLR0        30
#define SGR_CLR1        31
```

devices/console.h Page 2

```
#define SGR_CLR2        32
#define SGR_CLR3        33
#define SGR_CLR4        34
#define SGR_CLR5        35
#define SGR_CLR6        36
#define SGR_CLR7        37

#define SGR_CLR0BG      40
#define SGR_CLR1BG      41
#define SGR_CLR2BG      42
#define SGR_CLR3BG      43
#define SGR_CLR4BG      44
#define SGR_CLR5BG      45
#define SGR_CLR6BG      46
#define SGR_CLR7BG      47


/****** DSR parameters ******/

#define DSR_CPR         6

/****** CTC parameters ******/
#define CTC_HSETTAB     0
#define CTC_HCLRTAB     2
#define CTC_HCLRTABSALL 5

/****** TBC parameters ******/
#define TBC_HCLRTAB     0
#define TBC_HCLRTABSALL 3

/****** SM and RM parameters ******/
#define M_LNM   20      /* linefeed newline mode */
#define M_ASM   ">1"    /* auto scroll mode */
#define M_AWM   "?7"    /* auto wrap mode */

#endif  /* DEVICES_CONSOLE_H */
```

```
#ifndef DEVICES_CONUNIT_H
#define DEVICES_CONUNIT_H
/*
**      $Filename: devices/conunit.h $
**      $Release: 2.04 $
**      $Revision: 36.15 $
**      $Date: 90/11/20 $
**
**      Console device unit definitions
**
**      (C) Copyright 1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_TYPES_H
#include        "exec/types.h"
#endif

#ifndef EXEC_PORTS_H
#include        "exec/ports.h"
#endif

#ifndef DEVICES_CONSOLE_H
#include        "devices/console.h"
#endif

#ifndef DEVICES_KEYMAP_H
#include        "devices/keymap.h"
#endif

#ifndef DEVICES_INPUTEVENT_H
#include        "devices/inputevent.h"
#endif

/* ---- console unit numbers for OpenDevice() */
#define CONU_LIBRARY    -1      /* no unit, just fill in IO_DEVICE field */
#define CONU_STANDARD   0       /* standard unmapped console */

/* ---- New unit numbers for OpenDevice() - (V36) */

#define CONU_CHARMAP    1       /* bind character map to console */
#define CONU_SNIPMAP    3       /* bind character map w/ snip to console */

/* ---- New flag defines for OpenDevice() - (V37) */

#define CONFLAG_DEFAULT                 0
#define CONFLAG_NODRAW_ON_NEWSIZE       1


#define PMB_ASM         (M_LNM+1)       /* internal storage bit for AS flag */
#define PMB_AWM         (PMB_ASM+1)     /* internal storage bit for AW flag */
#define MAXTABS         80


struct  ConUnit {
    struct  MsgPort cu_MP;
    /* ---- read only variables */
    struct  Window *cu_Window;  /* intuition window bound to this unit */
    WORD    cu_XCP;             /* character position */
    WORD    cu_YCP;
    WORD    cu_XMax;            /* max character position */
    WORD    cu_YMax;
    WORD    cu_XRSize;          /* character raster size */
    WORD    cu_YRSize;
    WORD    cu_XROrigin;        /* raster origin */
    WORD    cu_YROrigin;
```

```
    WORD    cu_XRExtant;        /* raster maxima */
    WORD    cu_YRExtant;
    WORD    cu_XMinShrink;      /* smallest area intact from resize process */
    WORD    cu_YMinShrink;
    WORD    cu_XCCP;            /* cursor position */
    WORD    cu_YCCP;

    /* ---- read/write variables (writes must must be protected) */
    /* ---- storage for AskKeyMap and SetKeyMap */
    struct  KeyMap cu_KeyMapStruct;
    /* ---- tab stops */
    UWORD   cu_TabStops[MAXTABS]; /* 0 at start, 0xffff at end of list */

    /* ---- console rastport attributes */
    BYTE    cu_Mask;
    BYTE    cu_FgPen;
    BYTE    cu_BgPen;
    BYTE    cu_AOLPen;
    BYTE    cu_DrawMode;
    BYTE    cu_Obsolete1;       /* was cu_AreaPtSz -- not used in V36 */
    APTR    cu_Obsolete2;       /* was cu_AreaPtrn -- not used in V36 */
    UBYTE   cu_Minterms[8];     /* console minterms */
    struct  TextFont *cu_Font;
    UBYTE   cu_AlgoStyle;
    UBYTE   cu_TxFlags;
    UWORD   cu_TxHeight;
    UWORD   cu_TxWidth;
    UWORD   cu_TxBaseline;
    WORD    cu_TxSpacing;

    /* ---- console MODES and RAW EVENTS switches */
    UBYTE   cu_Modes[(PMB_AWM+7)/8];    /* one bit per mode */
    UBYTE   cu_RawEvents[(IECLASS_MAX+8)/8];
};

#endif  /* DEVICES_CONUNIT_H */
```

devices/gameport.h Page 1

```
#ifndef DEVICES_GAMEPORT_H
#define DEVICES_GAMEPORT_H
/*
**      $Filename: devices/gameport.h $
**      $Release: 2.04 $
**      $Revision: 36.1 $
**      $Date: 90/11/05 $
**
**      GamePort device command definitions
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_TYPES_H
#include        "exec/types.h"
#endif

#ifndef EXEC_IO_H
#include        "exec/io.h"
#endif

/******  GamePort commands ******/
#define  GPD_READEVENT      (CMD_NONSTD+0)
#define  GPD_ASKCTYPE       (CMD_NONSTD+1)
#define  GPD_SETCTYPE       (CMD_NONSTD+2)
#define  GPD_ASKTRIGGER     (CMD_NONSTD+3)
#define  GPD_SETTRIGGER     (CMD_NONSTD+4)

/******  GamePort structures ******/

/* gpt_Keys */
#define  GPTB_DOWNKEYS      0
#define  GPTF_DOWNKEYS      (1<<0)
#define  GPTB_UPKEYS        1
#define  GPTF_UPKEYS        (1<<1)

struct GamePortTrigger {
   UWORD gpt_Keys;          /* key transition triggers */
   UWORD gpt_Timeout;       /* time trigger (vertical blank units) */
   UWORD gpt_XDelta;        /* X distance trigger */
   UWORD gpt_YDelta;        /* Y distance trigger */
};

/****** Controller Types ******/
#define  GPCT_ALLOCATED    -1    /* allocated by another user */
#define  GPCT_NOCONTROLLER 0

#define  GPCT_MOUSE        1
#define  GPCT_RELJOYSTICK  2
#define  GPCT_ABSJOYSTICK  3


/****** Errors ******/
#define  GPDERR_SETCTYPE   1     /* this controller not valid at this time */

#endif  /* DEVICES_GAMEPORT_H */
```

devices/hardblocks.h Page 1

```
#ifndef DEVICES_HARDBLOCKS_H
#define DEVICES_HARDBLOCKS_H
/*
**      $Filename: devices/hardblocks.h $
**      $Release: 2.04 $
**      $Revision: 36.2 $
**      $Date: 90/11/06 $
**
**      File System identifier blocks for hard disks
**
**      (C) Copyright 1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_TYPES_H
#include "exec/types.h"
#endif /* EXEC_TYPES_H */

/*------------------------------------------------------------------
 *
 *      This file describes blocks of data that exist on a hard disk
 *      to describe that disk.  They are not generically accessable to
 *      the user as they do not appear on any DOS drive.  The blocks
 *      are tagged with a unique identifier, checksummed, and linked
 *      together.  The root of these blocks is the RigidDiskBlock.
 *
 *      The RigidDiskBlock must exist on the disk within the first
 *      RDB_LOCATION_LIMIT blocks.  This inhibits the use of the zero
 *      cylinder in an AmigaDOS partition: although it is strictly
 *      possible to store the RigidDiskBlock data in the reserved
 *      area of a partition, this practice is discouraged since the
 *      reserved blocks of a partition are overwritten by "Format",
 *      "Install", "DiskCopy", etc.  The recommended disk layout,
 *      then, is to use the first cylinder(s) to store all the drive
 *      data specified by these blocks: i.e. partition descriptions,
 *      file system load images, drive bad block maps, spare blocks, etc.
 *
 *      Though only 512 byte blocks are currently supported by the
 *      file system, this proposal tries to be forward-looking by
 *      making the block size explicit, and by using only the first
 *      256 bytes for all blocks but the LoadSeg data.
 *
 *-----------------------------------------------------------------*/

/*  NOTE
 *      optional block addresses below contain $ffffffff to indicate
 *      a NULL address, as zero is a valid address
 */
struct RigidDiskBlock {
    ULONG   rdb_ID;                 /* 4 character identifier */
    ULONG   rdb_SummedLongs;        /* size of this checksummed structure */
    LONG    rdb_ChkSum;             /* block checksum (longword sum to zero) */
    ULONG   rdb_HostID;             /* SCSI Target ID of host */
    ULONG   rdb_BlockBytes;         /* size of disk blocks */
    ULONG   rdb_Flags;              /* see below for defines */
    /* block list heads */
    ULONG   rdb_BadBlockList;       /* optional bad block list */
    ULONG   rdb_PartitionList;      /* optional first partition block */
    ULONG   rdb_FileSysHeaderList;  /* optional file system header block */
    ULONG   rdb_DriveInit;          /* optional drive-specific init code */
                                    /* DriveInit(lun,rdb,ior): "C" stk & d0/a0/a1 */
    ULONG   rdb_Reserved1[6];       /* set to $ffffffff */
    /* physical drive characteristics */
    ULONG   rdb_Cylinders;          /* number of drive cylinders */
    ULONG   rdb_Sectors;            /* sectors per track */
    ULONG   rdb_Heads;              /* number of drive heads */
    ULONG   rdb_Interleave;     /* interleave */
    ULONG   rdb_Park;           /* landing zone cylinder */
    ULONG   rdb_Reserved2[3];
    ULONG   rdb_WritePreComp;   /* starting cylinder: write precompensation */
    ULONG   rdb_ReducedWrite;   /* starting cylinder: reduced write current */
    ULONG   rdb_StepRate;       /* drive step rate */
    ULONG   rdb_Reserved3[5];
    /* logical drive characteristics */
    ULONG   rdb_RDBBlocksLo;    /* low block of range reserved for hardblocks */
    ULONG   rdb_RDBBlocksHi;    /* high block of range for these hardblocks */
    ULONG   rdb_LoCylinder;     /* low cylinder of partitionable disk area */
    ULONG   rdb_HiCylinder;     /* high cylinder of partitionable data area */
    ULONG   rdb_CylBlocks;      /* number of blocks available per cylinder */
    ULONG   rdb_AutoParkSeconds; /* zero for no auto park */
    ULONG   rdb_Reserved4[2];
    /* drive identification */
    char    rdb_DiskVendor[8];
    char    rdb_DiskProduct[16];
    char    rdb_DiskRevision[4];
    char    rdb_ControllerVendor[8];
    char    rdb_ControllerProduct[16];
    char    rdb_ControllerRevision[4];
    ULONG   rdb_Reserved5[10];
};

#define IDNAME_RIGIDDISK        0x5244534B      /* 'RDSK' */
#define RDB_LOCATION_LIMIT      16

#define RDBFB_LAST      0       /* no disks exist to be configured after */
#define RDBFF_LAST      0x01L   /*   this one on this controller */
#define RDBFB_LASTLUN   1       /* no LUNs exist to be configured greater */
#define RDBFF_LASTLUN   0x02L   /*   than this one at this SCSI Target ID */
#define RDBFB_LASTTID   2       /* no Target IDs exist to be configured */
#define RDBFF_LASTTID   0x04L   /*   greater than this one on this SCSI bus */
#define RDBFB_NORESELECT 3      /* don't bother trying to perform reselection */
#define RDBFF_NORESELECT 0x08L  /*   when talking to this drive */
#define RDBFB_DISKID    4       /* rdb_Disk... identification valid */
#define RDBFF_DISKID    0x10L
#define RDBFB_CTRLRID   5       /* rdb_Controller... identification valid */
#define RDBFF_CTRLRID   0x20L

/*------------------------------------------------------------------*/
struct BadBlockEntry {
    ULONG   bbe_BadBlock;       /* block number of bad block */
    ULONG   bbe_GoodBlock;      /* block number of replacement block */
};

struct BadBlockBlock {
    ULONG   bbb_ID;             /* 4 character identifier */
    ULONG   bbb_SummedLongs;    /* size of this checksummed structure */
    LONG    bbb_ChkSum;         /* block checksum (longword sum to zero) */
    ULONG   bbb_HostID;         /* SCSI Target ID of host */
    ULONG   bbb_Next;           /* block number of the next BadBlockBlock */
    ULONG   bbb_Reserved;
    struct BadBlockEntry bbb_BlockPairs[61]; /* bad block entry pairs */
    /* note [61] assumes 512 byte blocks */
};

#define IDNAME_BADBLOCK         0x42414442      /* 'BADB' */

/*------------------------------------------------------------------*/
struct PartitionBlock {
    ULONG   pb_ID;              /* 4 character identifier */
    ULONG   pb_SummedLongs;     /* size of this checksummed structure */
    LONG    pb_ChkSum;          /* block checksum (longword sum to zero) */
    ULONG   pb_HostID;          /* SCSI Target ID of host */
```

```
    ULONG   pb_Next;            /* block number of the next PartitionBlock */
    ULONG   pb_Flags;           /* see below for defines */
    ULONG   pb_Reserved1[2];
    ULONG   pb_DevFlags;        /* preferred flags for OpenDevice */
    UBYTE   pb_DriveName[32];   /* preferred DOS device name: BSTR form */
                                /* (not used if this name is in use) */
    ULONG   pb_Reserved2[15];   /* filler to 32 longwords */
    ULONG   pb_Environment[17]; /* environment vector for this partition */
    ULONG   pb_EReserved[15];   /* reserved for future environment vector */
};

#define IDNAME_PARTITION        0x50415254      /* 'PART' */

#define PBFB_BOOTABLE   0       /* this partition is intended to be bootable */
#define PBFF_BOOTABLE   1L      /*   (expected directories and files exist) */
#define PBFB_NOMOUNT    1       /* do not mount this partition (e.g. manually */
#define PBFF_NOMOUNT    2L      /*   mounted, but space reserved here) */

/*------------------------------------------------------------------*/
struct FileSysHeaderBlock {
    ULONG   fhb_ID;             /* 4 character identifier */
    ULONG   fhb_SummedLongs;    /* size of this checksummed structure */
    LONG    fhb_ChkSum;         /* block checksum (longword sum to zero) */
    ULONG   fhb_HostID;         /* SCSI Target ID of host */
    ULONG   fhb_Next;           /* block number of next FileSysHeaderBlock */
    ULONG   fhb_Flags;          /* see below for defines */
    ULONG   fhb_Reserved1[2];
    ULONG   fhb_DosType;        /* file system description: match this with */
                                /* partition environment's DE_DOSTYPE entry */
    ULONG   fhb_Version;        /* release version of this code */
    ULONG   fhb_PatchFlags;     /* bits set for those of the following that */
                                /*   need to be substituted into a standard */
                                /*   device node for this file system: e.g. */
                                /*   0x180 to substitute SegList & GlobalVec */
    ULONG   fhb_Type;           /* device node type: zero */
    ULONG   fhb_Task;           /* standard dos "task" field: zero */
    ULONG   fhb_Lock;           /* not used for devices: zero */
    ULONG   fhb_Handler;        /* filename to loadseg: zero placeholder */
    ULONG   fhb_StackSize;      /* stacksize to use when starting task */
    LONG    fhb_Priority;       /* task priority when starting task */
    LONG    fhb_Startup;        /* startup msg: zero placeholder */
    LONG    fhb_SegListBlocks;  /* first of linked list of LoadSegBlocks: */
                                /*   note that this entry requires some */
                                /*   processing before substitution */
    LONG    fhb_GlobalVec;      /* BCPL global vector when starting task */
    ULONG   fhb_Reserved2[23];  /* (those reserved by PatchFlags) */
    ULONG   fhb_Reserved3[21];
};

#define IDNAME_FILESYSHEADER    0x46534844      /* 'FSHD' */

/*------------------------------------------------------------------*/
struct LoadSegBlock {
    ULONG   lsb_ID;             /* 4 character identifier */
    ULONG   lsb_SummedLongs;    /* size of this checksummed structure */
    LONG    lsb_ChkSum;         /* block checksum (longword sum to zero) */
    ULONG   lsb_HostID;         /* SCSI Target ID of host */
    ULONG   lsb_Next;           /* block number of the next LoadSegBlock */
    ULONG   lsb_LoadData[123];  /* data for "loadseg" */
    /* note [123] assumes 512 byte blocks */
};

#define IDNAME_LOADSEG          0x4C534547      /* 'LSEG' */

#endif  /* DEVICES_HARDBLOCKS_H */
```

devices/input.h

```
#ifndef DEVICES_INPUT_H
#define DEVICES_INPUT_H
/*
**      $Filename: devices/input.h $
**      $Release: 2.04 $
**      $Revision: 36.0 $
**      $Date: 90/05/01 $
**
**      input device command definitions
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef     EXEC_IO_H
#include "exec/io.h"
#endif

#define  IND_ADDHANDLER     (CMD_NONSTD+0)
#define  IND_REMHANDLER     (CMD_NONSTD+1)
#define  IND_WRITEEVENT     (CMD_NONSTD+2)
#define  IND_SETTHRESH      (CMD_NONSTD+3)
#define  IND_SETPERIOD      (CMD_NONSTD+4)
#define  IND_SETMPORT       (CMD_NONSTD+5)
#define  IND_SETMTYPE       (CMD_NONSTD+6)
#define  IND_SETMTRIG       (CMD_NONSTD+7)

#endif  /* DEVICES_INPUT_H */
```

devices/inputevent.h

```
#ifndef DEVICES_INPUTEVENT_H
#define DEVICES_INPUTEVENT_H
/*
**      $Filename: devices/inputevent.h $
**      $Release: 2.04 $
**      $Revision: 36.7 $
**      $Date: 91/01/22 $
**
**      input event definitions
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef DEVICES_TIMER_H
#include "devices/timer.h"
#endif

/*----- constants --------------------------------------------------*/

/*  --- InputEvent.ie_Class --- */
/* A NOP input event */
#define IECLASS_NULL                    0x00
/* A raw keycode from the keyboard device */
#define IECLASS_RAWKEY                  0x01
/* The raw mouse report from the game port device */
#define IECLASS_RAWMOUSE                0x02
/* A private console event */
#define IECLASS_EVENT                   0x03
/* A Pointer Position report */
#define IECLASS_POINTERPOS              0x04
/* A timer event */
#define IECLASS_TIMER                   0x06
/* select button pressed down over a Gadget (address in ie_EventAddress) */
#define IECLASS_GADGETDOWN              0x07
/* select button released over the same Gadget (address in ie_EventAddress) */
#define IECLASS_GADGETUP                0x08
/* some Requester activity has taken place.  See Codes REQCLEAR and REQSET */
#define IECLASS_REQUESTER               0x09
/* this is a Menu Number transmission (Menu number is in ie_Code) */
#define IECLASS_MENULIST                0x0A
/* User has selected the active Window's Close Gadget */
#define IECLASS_CLOSEWINDOW             0x0B
/* this Window has a new size */
#define IECLASS_SIZEWINDOW              0x0C
/* the Window pointed to by ie_EventAddress needs to be refreshed */
#define IECLASS_REFRESHWINDOW           0x0D
/* new preferences are available */
#define IECLASS_NEWPREFS                0x0E
/* the disk has been removed */
#define IECLASS_DISKREMOVED             0x0F
/* the disk has been inserted */
#define IECLASS_DISKINSERTED            0x10
/* the window is about to be been made active */
#define IECLASS_ACTIVEWINDOW            0x11
/* the window is about to be made inactive */
#define IECLASS_INACTIVEWINDOW          0x12
/* extended-function pointer position report (V36) */
#define IECLASS_NEWPOINTERPOS           0x13
/* Help key report during Menu session (V36) */
#define IECLASS_MENUHELP                0x14
/* the Window has been modified with move, size, zoom, or change (V36) */
#define IECLASS_CHANGEWINDOW            0x15

/* the last class */
#define IECLASS_MAX                     0x15
```

```


/*  --- InputEvent.ie_SubClass --- */
/*  IECLASS_NEWPOINTERPOS */
/*      like IECLASS_POINTERPOS */
#define IESUBCLASS_COMPATIBLE   0x00
/*      ie_EventAddress points to struct IEPointerPixel */
#define IESUBCLASS_PIXEL        0x01
/*      ie_EventAddress points to struct IEPointerTablet */
#define IESUBCLASS_TABLET       0x02

/* pointed to by ie_EventAddress for IECLASS_NEWPOINTERPOS,
 * and IESUBCLASS_PIXEL.
 *
 * You specify a screen and pixel coordinates in that screen
 * at which you'd like the mouse to be positioned.
 * Intuition will try to oblige, but there will be restrictions
 * to positioning the pointer over offscreen pixels.
 *
 * IEQUALIFIER_RELATIVEMOUSE is supported for IESUBCLASS_PIXEL.
 */
struct IEPointerPixel   {
    struct Screen       *iepp_Screen;   /* pointer to an open screen */
    struct {                            /* pixel coordinates in iepp_Screen */
        WORD    X;
        WORD    Y;
    }                   iepp_Position;
};

/* pointed to by ie_EventAddress for IECLASS_NEWPOINTERPOS,
 * and IESUBCLASS_TABLET.
 *
 * You specify a range of values and a value within the range
 * independently for each of X and Y (the minimum value of
 * the ranges is always normalized to 0).
 *
 * Intuition will position the mouse proportionally within its
 * natural mouse position rectangle limits.
 *
 * IEQUALIFIER_RELATIVEMOUSE is not supported for IESUBCLASS_TABLET.
 */
struct IEPointerTablet  {
    struct {
        UWORD   X;
        UWORD   Y;
    }                   iept_Range;     /* 0 is min, these are max      */
    struct {
        UWORD   X;
        UWORD   Y;
    }                   iept_Value;     /* between 0 and iept_Range     */

    WORD                iept_Pressure;  /* -128 to 127 (unused, set to 0)  */
};



/*  --- InputEvent.ie_Code --- */
/*  IECLASS_RAWKEY */
#define IECODE_UP_PREFIX                0x80
#define IECODE_KEY_CODE_FIRST           0x00
#define IECODE_KEY_CODE_LAST            0x77
#define IECODE_COMM_CODE_FIRST          0x78
#define IECODE_COMM_CODE_LAST           0x7F

/*  IECLASS_ANSI */
#define IECODE_C0_FIRST                 0x00
```

```
#define IECODE_C0_LAST                  0x1F
#define IECODE_ASCII_FIRST              0x20
#define IECODE_ASCII_LAST               0x7E
#define IECODE_ASCII_DEL                0x7F
#define IECODE_C1_FIRST                 0x80
#define IECODE_C1_LAST                  0x9F
#define IECODE_LATIN1_FIRST             0xA0
#define IECODE_LATIN1_LAST              0xFF

/*  IECLASS_RAWMOUSE */
#define IECODE_LBUTTON                  0x68    /* also uses IECODE_UP_PREFIX */
#define IECODE_RBUTTON                  0x69
#define IECODE_MBUTTON                  0x6A
#define IECODE_NOBUTTON                 0xFF

/*  IECLASS_EVENT (V36) */
#define IECODE_NEWACTIVE                0x01    /* new active input window */
#define IECODE_NEWSIZE                  0x02    /* resize of window */
#define IECODE_REFRESH                  0x03    /* refresh of window */

/*  IECLASS_REQUESTER */
/*      broadcast when the first Requester (not subsequent ones) opens up in */
/*      the Window */
#define IECODE_REQSET                   0x01
/*      broadcast when the last Requester clears out of the Window */
#define IECODE_REQCLEAR                 0x00



/*  --- InputEvent.ie_Qualifier --- */
#define IEQUALIFIER_LSHIFT              0x0001
#define IEQUALIFIER_RSHIFT              0x0002
#define IEQUALIFIER_CAPSLOCK            0x0004
#define IEQUALIFIER_CONTROL             0x0008
#define IEQUALIFIER_LALT                0x0010
#define IEQUALIFIER_RALT                0x0020
#define IEQUALIFIER_LCOMMAND            0x0040
#define IEQUALIFIER_RCOMMAND            0x0080
#define IEQUALIFIER_NUMERICPAD          0x0100
#define IEQUALIFIER_REPEAT              0x0200
#define IEQUALIFIER_INTERRUPT           0x0400
#define IEQUALIFIER_MULTIBROADCAST      0x0800
#define IEQUALIFIER_MIDBUTTON           0x1000
#define IEQUALIFIER_RBUTTON             0x2000
#define IEQUALIFIER_LEFTBUTTON          0x4000
#define IEQUALIFIER_RELATIVEMOUSE       0x8000

#define IEQUALIFIERB_LSHIFT             0
#define IEQUALIFIERB_RSHIFT             1
#define IEQUALIFIERB_CAPSLOCK           2
#define IEQUALIFIERB_CONTROL            3
#define IEQUALIFIERB_LALT               4
#define IEQUALIFIERB_RALT               5
#define IEQUALIFIERB_LCOMMAND           6
#define IEQUALIFIERB_RCOMMAND           7
#define IEQUALIFIERB_NUMERICPAD         8
#define IEQUALIFIERB_REPEAT             9
#define IEQUALIFIERB_INTERRUPT          10
#define IEQUALIFIERB_MULTIBROADCAST     11
#define IEQUALIFIERB_MIDBUTTON          12
#define IEQUALIFIERB_RBUTTON            13
#define IEQUALIFIERB_LEFTBUTTON         14
#define IEQUALIFIERB_RELATIVEMOUSE      15

/*----- InputEvent ------------------------------------------------*/

```

devices/inputevent.h

```c
struct InputEvent {
    struct  InputEvent *ie_NextEvent;   /* the chronologically next event */
    UBYTE   ie_Class;                   /* the input event class */
    UBYTE   ie_SubClass;                /* optional subclass of the class */
    UWORD   ie_Code;                    /* the input event code */
    UWORD   ie_Qualifier;               /* qualifiers in effect for the event*/
    union {
        struct {
            WORD    ie_x;               /* the pointer position for the event*/
            WORD    ie_y;
        } ie_xy;
        APTR    ie_addr;                /* the event address */
        struct {
            UBYTE   ie_prev1DownCode;   /* previous down keys for dead */
            UBYTE   ie_prev1DownQual;   /*   key translation: the ie_Code */
            UBYTE   ie_prev2DownCode;   /*   & low byte of ie_Qualifier for */
            UBYTE   ie_prev2DownQual;   /*   last & second last down keys */
        } ie_dead;
    } ie_position;
    struct timeval ie_TimeStamp;        /* the system tick at the event */
};

#define ie_X                    ie_position.ie_xy.ie_x
#define ie_Y                    ie_position.ie_xy.ie_y
#define ie_EventAddress         ie_position.ie_addr
#define ie_Prev1DownCode        ie_position.ie_dead.ie_prev1DownCode
#define ie_Prev1DownQual        ie_position.ie_dead.ie_prev1DownQual
#define ie_Prev2DownCode        ie_position.ie_dead.ie_prev2DownCode
#define ie_Prev2DownQual        ie_position.ie_dead.ie_prev2DownQual

#endif  /* DEVICES_INPUTEVENT_H */
```

---

devices/keyboard.h

```c
#ifndef DEVICES_KEYBOARD_H
#define DEVICES_KEYBOARD_H
/*
**      $Filename: devices/keyboard.h $
**      $Release: 2.04 $
**      $Revision: 36.0 $
**      $Date: 90/05/01 $
**
**      Keyboard device command definitions
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef  EXEC_IO_H
#include "exec/io.h"
#endif

#define  KBD_READEVENT          (CMD_NONSTD+0)
#define  KBD_READMATRIX         (CMD_NONSTD+1)
#define  KBD_ADDRESETHANDLER    (CMD_NONSTD+2)
#define  KBD_REMRESETHANDLER    (CMD_NONSTD+3)
#define  KBD_RESETHANDLERDONE   (CMD_NONSTD+4)

#endif  /* DEVICES_KEYBOARD_H */
```

devices/keymap.h Page 1

```
#ifndef DEVICES_KEYMAP_H
#define DEVICES_KEYMAP_H
/*
**      $Filename: devices/keymap.h $
**      $Release: 2.04 $
**      $Revision: 36.3 $
**      $Date: 90/04/13 $
**
**      key map definitions for keymap.resource, keymap.library, and
**      console.device
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_NODES_H
#include        "exec/nodes.h"
#endif
#ifndef EXEC_LISTS_H
#include        "exec/lists.h"
#endif

struct   KeyMap {
    UBYTE   *km_LoKeyMapTypes;
    ULONG   *km_LoKeyMap;
    UBYTE   *km_LoCapsable;
    UBYTE   *km_LoRepeatable;
    UBYTE   *km_HiKeyMapTypes;
    ULONG   *km_HiKeyMap;
    UBYTE   *km_HiCapsable;
    UBYTE   *km_HiRepeatable;
};

struct  KeyMapNode {
    struct Node kn_Node;        /* including name of keymap */
    struct KeyMap kn_KeyMap;
};

/* the structure of keymap.resource */
struct  KeyMapResource {
    struct Node kr_Node;
    struct List kr_List;        /* a list of KeyMapNodes */
};

/* Key Map Types */
#define  KC_NOQUAL   0
#define  KC_VANILLA  7          /* note that SHIFT+ALT+CTRL is VANILLA */
#define  KCB_SHIFT   0
#define  KCF_SHIFT   0x01
#define  KCB_ALT     1
#define  KCF_ALT     0x02
#define  KCB_CONTROL 2
#define  KCF_CONTROL 0x04
#define  KCB_DOWNUP  3
#define  KCF_DOWNUP  0x08

#define  KCB_DEAD    5          /* may be dead or modified by dead key: */
#define  KCF_DEAD    0x20       /*   use dead prefix bytes              */

#define  KCB_STRING  6
#define  KCF_STRING  0x40

#define  KCB_NOP     7
#define  KCF_NOP     0x80


```

devices/keymap.h Page 2

```
/* Dead Prefix Bytes */
#define DPB_MOD 0
#define DPF_MOD 0x01
#define DPB_DEAD        3
#define DPF_DEAD        0x08

#define DP_2DINDEXMASK  0x0f    /* mask for index for 1st of two dead keys */
#define DP_2DFACSHIFT   4       /* shift for factor for 1st of two dead keys */

#endif  /* DEVICES_KEYMAP_H */
```

devices/narrator.h Page 1

```
#ifndef DEVICES_NARRATOR_H
#define DEVICES_NARRATOR_H
/*
**      $Filename: devices/narrator.h $
**      $Release: 2.04 $
**      $Revision: 1.7 $
**      $Date: 91/03/12 $
**
**      V37 Narrator device C language include file
**
**      Copyright 1990, 1991 Joseph Katz/Mark Barton.
**      All rights reserved.
**
**      This include file (narrator.h) may be freely distributed
**      as long as the above copyright notice remains intact.
**
*/


#ifndef EXEC_IO_H
#include "exec/io.h"
#endif


            /*          Device Options      */

#define NDB_NEWIORB     0       /* Use new extended IORB             */
#define NDB_WORDSYNC    1       /* Generate word sync messages       */
#define NDB_SYLSYNC     2       /* Generate syllable sync messages   */


#define NDF_NEWIORB     (1 << NDB_NEWIORB)
#define NDF_WORDSYNC    (1 << NDB_WORDSYNC)
#define NDF_SYLSYNC     (1 << NDB_SYLSYNC)




            /*          Error Codes         */

#define ND_NoMem        -2      /* Can't allocate memory             */
#define ND_NoAudLib     -3      /* Can't open audio device           */
#define ND_MakeBad      -4      /* Error in MakeLibrary call         */
#define ND_UnitErr      -5      /* Unit other than 0                 */
#define ND_CantAlloc    -6      /* Can't allocate audio channel(s)   */
#define ND_Unimpl       -7      /* Unimplemented command             */
#define ND_NoWrite      -8      /* Read for mouth without write first */
#define ND_Expunged     -9      /* Can't open, deferred expunge bit set */
#define ND_PhonErr     -20      /* Phoneme code spelling error
   */
#define ND_RateErr     -21      /* Rate out of bounds                */
#define ND_PitchErr    -22      /* Pitch out of bounds
   */
#define ND_SexErr      -23      /* Sex not valid                     */
#define ND_ModeErr     -24      /* Mode not valid                    */
#define ND_FreqErr     -25      /* Sampling frequency out of bounds  */
#define ND_VolErr      -26      /* Volume out of bounds         */
#define ND_DCentErr    -27      /* Degree of centralization out of bounds */
#define ND_CentPhonErr -28      /* Invalid central phon              */



            /* Input parameters and defaults */

#define DEFPITCH    110         /* Default pitch                     */
#define DEFRATE     150         /* Default speaking rate (wpm)
   */
```

devices/narrator.h Page 2

```
#define DEFVOL      64          /* Default volume (full)             */
#define DEFFREQ     22200       /* Default sampling frequency (Hz)   */
#define MALE        0           /* Male vocal tract                  */
#define FEMALE      1           /* Female vocal tract                */
#define NATURALF0   0           /* Natural pitch contours            */
#define ROBOTICF0   1           /* Monotone pitch                    */
#define MANUALF0    2           /* Manual setting of pitch contours  */
#define DEFSEX      MALE        /* Default sex
  */
#define DEFMODE     NATURALF0   /* Default mode                 */
#define DEFARTIC    100         /* 100% articulation (normal)        */
#define DEFCENTRAL  0           /* No centralization                 */
#define DEFF0PERT   0           /* No F0 Perturbation                */
#define DEFF0ENTHUS 32          /* Default F0 enthusiasm (in 32nds)  */
#define DEFPRIORITY 100         /* Default speaking priority         */


                        /*      Parameter bounds        */

#define MINRATE     40          /* Minimum speaking rate             */
#define MAXRATE     400         /* Maximum speaking rate             */
#define MINPITCH    65          /* Minimum pitch                     */
#define MAXPITCH    320         /* Maximum pitch                     */
#define MINFREQ     5000        /* Minimum sampling frequency        */
#define MAXFREQ     28000       /* Maximum sampling frequency        */
#define MINVOL      0           /* Minimum volume                    */
#define MAXVOL      64          /* Maximum volume                    */
#define MINCENT      0          /* Minimum degree of centralization  */
#define MAXCENT    100          /* Maximum degree of centralization  */


                /*    Standard Write request    */

struct narrator_rb {
        struct IOStdReq  message;       /* Standard IORB               */
        UWORD   rate;                   /* Speaking rate (words/minute) */
        UWORD   pitch;                  /* Baseline pitch in Hertz
  */
        UWORD   mode;                   /* Pitch mode                  */
        UWORD   sex;                    /* Sex of voice                */
        UBYTE   *ch_masks;              /* Pointer to audio alloc maps */
        UWORD   nm_masks;               /* Number of audio alloc maps  */
        UWORD   volume;                 /* Volume. 0 (off) thru 64     */
        UWORD   sampfreq;               /* Audio sampling freq
  */
        UBYTE   mouths;                 /* If non-zero, generate mouths */
        UBYTE   chanmask;               /* Which ch mask used (internal)*/
        UBYTE   numchan;                /* Num ch masks used (internal) */
        UBYTE   flags;                  /* New feature flags           */
        UBYTE   F0enthusiasm;           /* F0 excursion factor         */
        UBYTE   F0perturb;              /* Amount of F0 perturbation   */
        BYTE    F1adj;                  /* F1 adjustment in M-15% steps */
        BYTE    F2adj;                          /* F2 adjustment in M-15% steps
  */
        BYTE    F3adj;                  /* F3 adjustment in M-15% steps */
        BYTE    A1adj;                  /* A1 adjustment in decibels   */
        BYTE    A2adj;                  /* A2 adjustment in decibels   */
        BYTE    A3adj;                  /* A3 adjustment in decibels   */
        UBYTE   articulate;             /* Transition time multiplier  */
        UBYTE   centralize;             /* Degree of vowel centralization */
        char    *centphon;              /* Pointer to central ASCII phon */
        BYTE    AVbias;                 /* AV bias                     */
        BYTE    AFbias;                 /* AF bias                     */
        BYTE    priority;               /* Priority while speaking     */
        BYTE    pad1;                   /* For alignment               */
    };
```

devices/narrator.h

```



                /*    Standard Read request     */

struct mouth_rb {
        struct  narrator_rb voice;      /* Speech IORB                  */
        UBYTE   width;                  /* Width (returned value)       */
        UBYTE   height;                 /* Height (returned value)      */
        UBYTE   shape;                  /* Internal use, do not modify  */
        UBYTE   sync;                   /* Returned sync events         */
        };



#endif  /* DEVICES_NARRATOR_H */
```

devices/parallel.h

```
#ifndef DEVICES_PARALLEL_H
#define DEVICES_PARALLEL_H
/*
**      $Filename: devices/parallel.h $
**      $Release: 2.04 $
**      $Revision: 36.1 $
**      $Date: 90/05/10 $
**
**      parallel.device I/O request structure information
**
**      (C) Copyright 1985,1988,1989,1990 Commodore Amiga Inc.
**          All rights reserved.
*/

#ifndef   EXEC_IO_H
#include "exec/io.h"
#endif   /* !EXEC_IO_H */

 struct  IOPArray {
        ULONG PTermArray0;
        ULONG PTermArray1;
};

/******************************************************************/
/* CAUTION !!  IF YOU ACCESS the parallel.device, you MUST (!!!!) use
   an IOExtPar-sized structure or you may overlay innocent memory !! */
/******************************************************************/

 struct   IOExtPar {
        struct   IOStdReq IOPar;

/*      STRUCT   MsgNode
*   0   APTR     Succ
*   4   APTR     Pred
*   8   UBYTE    Type
*   9   UBYTE    Pri
*   A   APTR     Name
*   E   APTR     ReplyPort
*  12   UWORD    MNLength
*      STRUCT   IOExt
*  14   APTR     io_Device
*  18   APTR     io_Unit
*  1C   UWORD    io_Command
*  1E   UBYTE    io_Flags
*  1F   UBYTE    io_Error
*      STRUCT   IOStdExt
*  20   ULONG    io_Actual
*  24   ULONG    io_Length
*  28   APTR     io_Data
*  2C   ULONG    io_Offset
*  30 */
        ULONG   io_PExtFlags;     /* (not used) flag extension area */
        UBYTE   io_Status;        /* status of parallel port and registers */
        UBYTE   io_ParFlags;      /* see PARFLAGS bit definitions below */
        struct  IOPArray io_PTermArray; /* termination character array */
};

#define PARB_SHARED     5          /* ParFlags non-exclusive access bit */
#define PARF_SHARED     (1<<5)     /*    "     non-exclusive access mask */
#define PARB_SLOWMODE   4          /*    "     slow printer bit */
#define PARF_SLOWMODE   (1<<4)     /*    "     slow printer mask */
#define PARB_FASTMODE   3          /*    "     fast I/O mode selected bit */
#define PARF_FASTMODE   (1<<3)     /*    "     fast I/O mode selected mask */
#define PARB_RAD_BOOGIE 3          /*    "     for backward compatibility */
#define PARF_RAD_BOOGIE (1<<3)     /*    "     for backward compatibility */

```

devices/parallel.h

```
#define PARB_ACKMODE    2          /*    "     ACK interrupt handshake bit */
#define PARF_ACKMODE    (1<<2)     /*    "     ACK interrupt handshake mask */

#define PARB_EOFMODE    1          /*    "     EOF mode enabled bit */
#define PARF_EOFMODE    (1<<1)     /*    "     EOF mode enabled mask */

#define IOPARB_QUEUED   6          /* IO_FLAGS rqst-queued bit */
#define IOPARF_QUEUED   (1<<6)     /*    "     rqst-queued mask */
#define IOPARB_ABORT    5          /*    "     rqst-aborted bit */
#define IOPARF_ABORT    (1<<5)     /*    "     rqst-aborted mask */
#define IOPARB_ACTIVE   4          /*    "     rqst-qued-or-current bit */
#define IOPARF_ACTIVE   (1<<4)     /*    "     rqst-qued-or-current mask */
#define IOPTB_RWDIR     3          /* IO_STATUS read=0,write=1 bit */
#define IOPTF_RWDIR     (1<<3)     /*    "     read=0,write=1 mask */
#define IOPTB_PARSEL    2          /*    "     printer selected on the A1000 */
#define IOPTF_PARSEL    (1<<2)     /* printer selected & serial "Ring Indicator"
                                      on the A500 & A2000.  Be careful when
                                      making cables */
#define IOPTB_PAPEROUT 1           /*    "     paper out bit */
#define IOPTF_PAPEROUT (1<<1)      /*    "     paper out mask */
#define IOPTB_PARBUSY  0           /*    "     printer in busy toggle bit */
#define IOPTF_PARBUSY  (1<<0)      /*    "     printer in busy toggle mask */
/* Note: previous versions of this include files had bits 0 and 2 swapped */

#define PARALLELNAME             "parallel.device"

#define PDCMD_QUERY              (CMD_NONSTD)
#define PDCMD_SETPARAMS (CMD_NONSTD+1)

#define ParErr_DevBusy                 1
#define ParErr_BufTooBig        2
#define ParErr_InvParam 3
#define ParErr_LineErr          4
#define ParErr_NotOpen          5
#define ParErr_PortReset        6
#define ParErr_InitErr                 7

#endif  /* DEVICES_PARALLEL_H */
```

devices/printer.h

```
#ifndef     DEVICES_PRINTER_H
#define     DEVICES_PRINTER_H
/*
**      $Filename: devices/printer.h $
**      $Release: 2.04 $
**      $Revision: 1.7 $
**      $Date: 90/07/26 $
**
**      printer.device structure definitions
**
**      (C) Copyright 1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef  EXEC_TYPES_H
#include "exec/types.h"
#endif

#ifndef  EXEC_NODES_H
#include "exec/nodes.h"
#endif

#ifndef  EXEC_LISTS_H
#include "exec/lists.h"
#endif

#ifndef  EXEC_PORTS_H
#include "exec/ports.h"
#endif

#define  PRD_RAWWRITE       (CMD_NONSTD+0)
#define  PRD_PRTCOMMAND     (CMD_NONSTD+1)
#define  PRD_DUMPRPORT      (CMD_NONSTD+2)
#define  PRD_QUERY          (CMD_NONSTD+3)

/* printer command definitions */

#define aRIS     0  /* ESCc  reset                      ISO */
#define aRIN     1  /* ESC#1 initialize         +++ */
#define aIND     2  /* ESCD  lf                 ISO */
#define aNEL     3  /* ESCE  return,lf                  ISO */
#define aRI      4  /* ESCM  reverse lf         ISO */

#define aSGR0    5  /* ESC[0m normal char set           ISO */
#define aSGR3    6  /* ESC[3m italics on                ISO */
#define aSGR23   7  /* ESC[23m italics off              ISO */
#define aSGR4    8  /* ESC[4m underline on              ISO */
#define aSGR24   9  /* ESC[24m underline off            ISO */
#define aSGR1   10  /* ESC[1m boldface on               ISO */
#define aSGR22  11  /* ESC[22m boldface off             ISO */
#define aSFC    12  /* SGR30-39  set foreground color ISO */
#define aSBC    13  /* SGR40-49  set background color ISO */

#define aSHORP0 14  /* ESC[0w normal pitch              DEC */
#define aSHORP2 15  /* ESC[2w elite on                  DEC */
#define aSHORP1 16  /* ESC[1w elite off         DEC */
#define aSHORP4 17  /* ESC[4w condensed fine on         DEC */
#define aSHORP3 18  /* ESC[3w condensed off             DEC */
#define aSHORP6 19  /* ESC[6w enlarged on               DEC */
#define aSHORP5 20  /* ESC[5w enlarged off              DEC */

#define aDEN6   21  /* ESC[6"z shadow print on          DEC (sort of) */
#define aDEN5   22  /* ESC[5"z shadow print off         DEC */
#define aDEN4   23  /* ESC[4"z doublestrike on          DEC */
#define aDEN3   24  /* ESC[3"z doublestrike off         DEC */
#define aDEN2   25  /* ESC[2"z  NLQ on                  DEC */
```

```
#define aDEN1   26  /* ESC[1"z  NLQ off        DEC */

#define aSUS2   27  /* ESC[2v superscript on          +++ */
#define aSUS1   28  /* ESC[1v superscript off         +++ */
#define aSUS4   29  /* ESC[4v subscript on            +++ */
#define aSUS3   30  /* ESC[3v subscript off           +++ */
#define aSUS0   31  /* ESC[0v normalize the line      +++ */
#define aPLU    32  /* ESCL  partial line up          ISO */
#define aPLD    33  /* ESCK  partial line down        ISO */

#define aFNT0   34  /* ESC(B US char set        or Typeface  0 (default) */
#define aFNT1   35  /* ESC(R French char set    or Typeface  1 */
#define aFNT2   36  /* ESC(K German char set    or Typeface  2 */
#define aFNT3   37  /* ESC(A UK char set        or Typeface  3 */
#define aFNT4   38  /* ESC(E Danish I char set  or Typeface  4 */
#define aFNT5   39  /* ESC(H Sweden char set    or Typeface  5 */
#define aFNT6   40  /* ESC(Y Italian char set   or Typeface  6 */
#define aFNT7   41  /* ESC(Z Spanish char set   or Typeface  7 */
#define aFNT8   42  /* ESC(J Japanese char set  or Typeface  8 */
#define aFNT9   43  /* ESC(6 Norweign char set  or Typeface  9 */
#define aFNT10  44  /* ESC(C Danish II char set or Typeface 10 */

/*
        Suggested typefaces are:

         0 - default typeface.
         1 - Line Printer or equiv.
         2 - Pica or equiv.
         3 - Elite or equiv.
         4 - Helvetica or equiv.
         5 - Times Roman or equiv.
         6 - Gothic or equiv.
         7 - Script or equiv.
         8 - Prestige or equiv.
         9 - Caslon or equiv.
        10 - Orator or equiv.
*/

#define aPROP2  45  /* ESC[2p  proportional on        +++ */
#define aPROP1  46  /* ESC[1p  proportional off       +++ */
#define aPROP0  47  /* ESC[0p  proportional clear     +++ */
#define aTSS    48  /* ESC[n E set proportional offset ISO */
#define aJFY5   49  /* ESC[5 F auto left justify      ISO */
#define aJFY7   50  /* ESC[7 F auto right justify     ISO */
#define aJFY6   51  /* ESC[6 F auto full justify      ISO */
#define aJFY0   52  /* ESC[0 F auto justify off       ISO */
#define aJFY3   53  /* ESC[3 F letter space (justify) ISO (special) */
#define aJFY1   54  /* ESC[1 F word fill(auto center) ISO (special) */

#define aVERP0  55  /* ESC[0z  1/8" line spacing      +++ */
#define aVERP1  56  /* ESC[1z  1/6" line spacing      +++ */
#define aSLPP   57  /* ESC[nt  set form length n      DEC */
#define aPERF   58  /* ESC[nq  perf skip n (n>0)      +++ */
#define aPERF0  59  /* ESC[0q  perf skip off          +++ */

#define aLMS    60  /* ESC#9   Left margin set        +++ */
#define aRMS    61  /* ESC#0   Right margin set       +++ */
#define aTMS    62  /* ESC#8   Top margin set         +++ */
#define aBMS    63  /* ESC#2   Bottom marg set        +++ */
#define aSTBM   64  /* ESC[Pn1;Pn2r  T&B margins      DEC */
#define aSLRM   65  /* ESC[Pn1;Pn2s  L&R margin       DEC */
#define aCAM    66  /* ESC#3   Clear margins          +++ */

#define aHTS    67  /* ESCH    Set horiz tab          ISO */
#define aVTS    68  /* ESCJ    Set vertical tabs      ISO */
#define aTBC0   69  /* ESC[0g  Clr horiz tab          ISO */
```

```
#define aTBC3   70  /* ESC[3g  Clear all h tab        ISO */
#define aTBC1   71  /* ESC[1g  Clr vertical tabs      ISO */
#define aTBC4   72  /* ESC[4g  Clr all v tabs         ISO */
#define aTBCALL 73  /* ESC#4   Clr all h & v tabs     +++ */
#define aTBSALL 74  /* ESC#5   Set default tabs       +++ */
#define aEXTEND 75  /* ESC[Pn"x extended commands     +++ */

#define aRAW    76      /* ESC[Pn"r     Next 'Pn' chars are raw +++ */

struct IOPrtCmdReq {
    struct  Message io_Message;
    struct  Device  *io_Device;     /* device node pointer  */
    struct  Unit    *io_Unit;       /* unit (driver private)*/
    UWORD   io_Command;     /* device command */
    UBYTE   io_Flags;
    BYTE    io_Error;               /* error or warning num */
    UWORD   io_PrtCommand;          /* printer command */
    UBYTE   io_Parm0;               /* first command parameter */
    UBYTE   io_Parm1;               /* second command parameter */
    UBYTE   io_Parm2;               /* third command parameter */
    UBYTE   io_Parm3;               /* fourth command parameter */
};

struct IODRPReq {
    struct  Message io_Message;
    struct  Device  *io_Device;     /* device node pointer  */
    struct  Unit    *io_Unit;       /* unit (driver private)*/
    UWORD   io_Command;     /* device command */
    UBYTE   io_Flags;
    BYTE    io_Error;               /* error or warning num */
    struct  RastPort *io_RastPort;  /* raster port */
    struct  ColorMap *io_ColorMap;  /* color map */
    ULONG   io_Modes;               /* graphics viewport modes */
    UWORD   io_SrcX;                /* source x origin */
    UWORD   io_SrcY;                /* source y origin */
    UWORD   io_SrcWidth;            /* source x width */
    UWORD   io_SrcHeight;           /* source x height */
    LONG    io_DestCols;            /* destination x width */
    LONG    io_DestRows;            /* destination y height */
    UWORD   io_Special;     /* option flags */
};

#define SPECIAL_MILCOLS         0x0001  /* DestCols specified in 1/1000" */
#define SPECIAL_MILROWS         0x0002  /* DestRows specified in 1/1000" */
#define SPECIAL_FULLCOLS        0x0004  /* make DestCols maximum possible */
#define SPECIAL_FULLROWS        0x0008  /* make DestRows maximum possible */
#define SPECIAL_FRACCOLS        0x0010  /* DestCols is fraction of FULLCOLS */
#define SPECIAL_FRACROWS        0x0020  /* DestRows is fraction of FULLROWS */
#define SPECIAL_CENTER          0x0040  /* center image on paper */
#define SPECIAL_ASPECT          0x0080  /* ensure correct aspect ratio */
#define SPECIAL_DENSITY1        0x0100  /* lowest resolution (dpi) */
#define SPECIAL_DENSITY2        0x0200  /* next res */
#define SPECIAL_DENSITY3        0x0300  /* next res */
#define SPECIAL_DENSITY4        0x0400  /* next res */
#define SPECIAL_DENSITY5        0x0500  /* next res */
#define SPECIAL_DENSITY6        0x0600  /* next res */
#define SPECIAL_DENSITY7        0x0700  /* highest res */
#define SPECIAL_NOFORMFEED      0x0800  /* don't eject paper on gfx prints */
#define SPECIAL_TRUSTME         0x1000  /* don't reset on gfx prints */
/*
        Compute print size, set 'io_DestCols' and 'io_DestRows' in the calling
        program's 'IODRPReq' structure and exit, DON'T PRINT.  This allows the
        calling program to see what the final print size would be in printer
        pixels.  Note that it modifies the 'io_DestCols' and 'io_DestRows'
        fields of your 'IODRPReq' structure.  Also, set the print density and
        update the 'MaxXDots', 'MaxYDots', 'XDotsInch', and 'YDotsInch' fields
```

devices/printer.h

```
        of the 'PrinterExtendedData' structure.
*/
#define SPECIAL_NOPRINT         0x2000  /* see above */

#define PDERR_NOERR             0       /* clean exit, no errors */
#define PDERR_CANCEL            1       /* user cancelled print */
#define PDERR_NOTGRAPHICS       2       /* printer cannot output graphics */
#define PDERR_INVERTHAM         3       /* OBSOLETE */
#define PDERR_BADDIMENSION      4       /* print dimensions illegal */
#define PDERR_DIMENSIONOVFLOW   5       /* OBSOLETE */
#define PDERR_INTERNALMEMORY    6       /* no memory for internal variables */
#define PDERR_BUFFERMEMORY      7       /* no memory for print buffer */
/*
        Note : this is an internal error that can be returned from the render
        function to the printer device.  It is NEVER returned to the user.
        If the printer device sees this error it converts it 'PDERR_NOERR'
        and exits gracefully.  Refer to the document on
        'How to Write a Graphics Printer Driver' for more info.
*/
#define PDERR_TOOKCONTROL       8       /* Took control in case 0 of render */

/* internal use */
#define SPECIAL_DENSITYMASK     0x0700  /* masks out density values */
#define SPECIAL_DIMENSIONSMASK \
        (SPECIAL_MILCOLS|SPECIAL_MILROWS|SPECIAL_FULLCOLS|SPECIAL_FULLROWS\
        |SPECIAL_FRACCOLS|SPECIAL_FRACROWS|SPECIAL_ASPECT)
#endif
```

devices/prtbase.h

```
#ifndef  DEVICES_PRTBASE_H
#define  DEVICES_PRTBASE_H
/*
**      $Filename: devices/prtbase.h $
**      $Release: 2.04 $
**      $Revision: 1.10 $
**      $Date: 90/11/02 $
**
**      printer.device base structure definitions
**
**      (C) Copyright 1987, 1988,1989, 1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef  EXEC_TYPES_H
#include "exec/types.h"
#endif
#ifndef  EXEC_NODES_H
#include "exec/nodes.h"
#endif
#ifndef  EXEC_LISTS_H
#include "exec/lists.h"
#endif
#ifndef  EXEC_PORTS_H
#include "exec/ports.h"
#endif
#ifndef  EXEC_LIBRARIES_H
#include "exec/libraries.h"
#endif
#ifndef  EXEC_TASKS_H
#include "exec/tasks.h"
#endif

#ifndef  DEVICES_PARALLEL_H
#include "devices/parallel.h"
#endif
#ifndef  DEVICES_SERIAL_H
#include "devices/serial.h"
#endif
#ifndef  DEVICES_TIMER_H
#include "devices/timer.h"
#endif
#ifndef  LIBRARIES_DOSEXTENS_H
#include "libraries/dosextens.h"
#endif
#ifndef  INTUITION_INTUITION_H
#include "intuition/intuition.h"
#endif


struct DeviceData {
    struct Library dd_Device; /* standard library node */
    APTR dd_Segment;          /* A0 when initialized */
    APTR dd_ExecBase;         /* A6 for exec */
    APTR dd_CmdVectors;       /* command table for device commands */
    APTR dd_CmdBytes;         /* bytes describing which command queue */
    UWORD   dd_NumCommands;   /* the number of commands supported */
};

#define P_OLDSTKSIZE    0x0800  /* stack size for child task (OBSOLETE) */
#define P_STKSIZE       0x1000  /* stack size for child task */
#define P_BUFSIZE       256     /* size of internal buffers for text i/o */
#define P_SAFESIZE      128     /* safety margin for text output buffer */

struct   PrinterData {
        struct DeviceData pd_Device;
        struct MsgPort pd_Unit; /* the one and only unit */
        BPTR pd_PrinterSegment; /* the printer specific segment */
        UWORD pd_PrinterType;   /* the segment printer type */
                                /* the segment data structure */
        struct PrinterSegment *pd_SegmentData;
        UBYTE *pd_PrintBuf;     /* the raster print buffer */
        int (*pd_PWrite)();     /* the write function */
        int (*pd_PBothReady)(); /* write function's done */
        union {                 /* port I/O request 0 */
                struct IOExtPar pd_p0;
                struct IOExtSer pd_s0;
        } pd_ior0;

#define  pd_PIOR0 pd_ior0.pd_p0
#define  pd_SIOR0 pd_ior0.pd_s0

        union {                 /*   and 1 for double buffering */
                struct IOExtPar pd_p1;
                struct IOExtSer pd_s1;
        } pd_ior1;

#define  pd_PIOR1 pd_ior1.pd_p1
#define  pd_SIOR1 pd_ior1.pd_s1

        struct timerequest pd_TIOR;     /* timer I/O request */
        struct MsgPort pd_IORPort;      /* and message reply port */
        struct Task pd_TC;              /* write task */
        UBYTE pd_OldStk[P_OLDSTKSIZE];  /* and stack space (OBSOLETE) */
        UBYTE pd_Flags;                 /* device flags */
        UBYTE pd_pad;                   /* padding */
        struct Preferences pd_Preferences;      /* the latest preferences */
        UBYTE pd_PWaitEnabled;          /* wait function switch */
        /* new fields for V2.0 */
        UBYTE pd_Flags1;                /* padding */
        UBYTE pd_Stk[P_STKSIZE];        /* stack space */
};

/* Printer Class */
#define PPCB_GFX        0       /* graphics (bit position) */
#define PPCF_GFX        0x1     /* graphics (and/or flag) */
#define PPCB_COLOR      1       /* color (bit position) */
#define PPCF_COLOR      0x2     /* color (and/or flag) */

#define PPC_BWALPHA     0x00    /* black&white alphanumerics */
#define PPC_BWGFX       0x01    /* black&white graphics */
#define PPC_COLORALPHA  0x02    /* color alphanumerics */
#define PPC_COLORGFX    0x03    /* color graphics */

/* Color Class */
#define PCC_BW          0x01    /* black&white only */
#define PCC_YMC         0x02    /* yellow/magenta/cyan only */
#define PCC_YMC_BW      0x03    /* yellow/magenta/cyan or black&white */
#define PCC_YMCB        0x04    /* yellow/magenta/cyan/black */
#define PCC_4COLOR      0x04    /* a flag for YMCB and BGRW */
#define PCC_ADDITIVE    0x08    /* not ymcb but blue/green/red/white */
#define PCC_WB          0x09    /* black&white only, 0 == BLACK */
#define PCC_BGR         0x0A    /* blue/green/red */
#define PCC_BGR_WB      0x0B    /* blue/green/red or black&white */
#define PCC_BGRW        0x0C    /* blue/green/red/white */
/*
        The picture must be scanned once for each color component, as the
        printer can only define one color at a time.  ie. If 'PCC_YMC' then
        first pass sends all 'Y' info to printer, second pass sends all 'M'
        info, and third pass sends all C info to printer.  The CalComp
        PlotMaster is an example of this type of printer.
*/
```

```
#define PCC_MULTI_PASS  0x10    /* see explanation above */

struct PrinterExtendedData {
        char    *ped_PrinterName;       /* printer name, null terminated */
        VOID    (*ped_Init)();          /* called after LoadSeg */
        VOID    (*ped_Expunge)();       /* called before UnLoadSeg */
        int     (*ped_Open)();          /* called at OpenDevice */
        VOID    (*ped_Close)();         /* called at CloseDevice */
        UBYTE   ped_PrinterClass;       /* printer class */
        UBYTE   ped_ColorClass;         /* color class */
        UBYTE   ped_MaxColumns;         /* number of print columns available */
        UBYTE   ped_NumCharSets;        /* number of character sets */
        UWORD   ped_NumRows;            /* number of 'pins' in print head */
        ULONG   ped_MaxXDots;           /* number of dots max in a raster dump */
        ULONG   ped_MaxYDots;           /* number of dots max in a raster dump */
        UWORD   ped_XDotsInch;          /* horizontal dot density */
        UWORD   ped_YDotsInch;          /* vertical dot density */
        char    ***ped_Commands;        /* printer text command table */
        int     (*ped_DoSpecial)();     /* special command handler */
        int     (*ped_Render)();        /* raster render function */
        LONG    ped_TimeoutSecs;        /* good write timeout */
        /* the following only exists if the segment version is >= 33 */
        char    **ped_8BitChars;        /* conv. strings for the extended font */
        LONG    ped_PrintMode;          /* set if text printed, otherwise 0 */
        /* the following only exists if the segment version is >= 34 */
        /* ptr to conversion function for all chars */
        int     (*ped_ConvFunc)();
};

struct PrinterSegment {
    ULONG   ps_NextSegment;         /* (actually a BPTR) */
    ULONG   ps_runAlert;            /* MOVEQ #0,D0 : RTS */
    UWORD   ps_Version;  /* segment version */
    UWORD   ps_Revision;            /* segment revision */
    struct  PrinterExtendedData ps_PED;    /* printer extended data */
};
#endif
```

devices/prtgfx.h Page 1

```
#ifndef DEVICES_PRTGFX_H
#define DEVICES_PRTGFX_H
/*
**      $Filename: devices/prtgfx.h $
**      $Release: 2.04 $
**      $Revision: 1.12 $
**      $Date: 90/07/26 $
**
**      printer.device structure definitions
**
**      (C) Copyright 1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef  EXEC_TYPES_H
#include "exec/types.h"
#endif

#define PCMYELLOW       0               /* byte index for yellow */
#define PCMMAGENTA      1               /* byte index for magenta */
#define PCMCYAN         2               /* byte index for cyan */
#define PCMBLACK        3               /* byte index for black */
#define PCMBLUE         PCMYELLOW       /* byte index for blue */
#define PCMGREEN        PCMMAGENTA      /* byte index for green */
#define PCMRED          PCMCYAN         /* byte index for red */
#define PCMWHITE        PCMBLACK        /* byte index for white */

union colorEntry {
        ULONG colorLong;        /* quick access to all of YMCB */
        UBYTE colorByte[4];     /* 1 entry for each of YMCB */
        BYTE colorSByte[4];     /* ditto (except signed) */
};

struct PrtInfo { /* printer info */
        int (*pi_render)();             /* PRIVATE - DO NOT USE! */
        struct RastPort *pi_rp;         /* PRIVATE - DO NOT USE! */
        struct RastPort *pi_temprp;     /* PRIVATE - DO NOT USE! */
        UWORD *pi_RowBuf;               /* PRIVATE - DO NOT USE! */
        UWORD *pi_HamBuf;               /* PRIVATE - DO NOT USE! */
        union colorEntry *pi_ColorMap;  /* PRIVATE - DO NOT USE! */
        union colorEntry *pi_ColorInt;  /* color intensities for entire row */
        union colorEntry *pi_HamInt;    /* PRIVATE - DO NOT USE! */
        union colorEntry *pi_Dest1Int;  /* PRIVATE - DO NOT USE! */
        union colorEntry *pi_Dest2Int;  /* PRIVATE - DO NOT USE! */
        UWORD *pi_ScaleX;               /* array of scale values for X */
        UWORD *pi_ScaleXAlt;            /* PRIVATE - DO NOT USE! */
        UBYTE *pi_dmatrix;              /* pointer to dither matrix */
        UWORD *pi_TopBuf;               /* PRIVATE - DO NOT USE! */
        UWORD *pi_BotBuf;               /* PRIVATE - DO NOT USE! */

        UWORD pi_RowBufSize;            /* PRIVATE - DO NOT USE! */
        UWORD pi_HamBufSize;            /* PRIVATE - DO NOT USE! */
        UWORD pi_ColorMapSize;          /* PRIVATE - DO NOT USE! */
        UWORD pi_ColorIntSize;          /* PRIVATE - DO NOT USE! */
        UWORD pi_HamIntSize;            /* PRIVATE - DO NOT USE! */
        UWORD pi_Dest1IntSize;          /* PRIVATE - DO NOT USE! */
        UWORD pi_Dest2IntSize;          /* PRIVATE - DO NOT USE! */
        UWORD pi_ScaleXSize;            /* PRIVATE - DO NOT USE! */
        UWORD pi_ScaleXAltSize;         /* PRIVATE - DO NOT USE! */

        UWORD pi_PrefsFlags;            /* PRIVATE - DO NOT USE! */
        ULONG pi_special;               /* PRIVATE - DO NOT USE! */
        UWORD pi_xstart;                /* PRIVATE - DO NOT USE! */
        UWORD pi_ystart;                /* PRIVATE - DO NOT USE! */
        UWORD pi_width;                 /* source width (in pixels) */
        UWORD pi_height;                /* PRIVATE - DO NOT USE! */
```

devices/prtgfx.h Page 2

```
        ULONG pi_pc;                    /* PRIVATE - DO NOT USE! */
        ULONG pi_pr;                    /* PRIVATE - DO NOT USE! */
        UWORD pi_ymult;                 /* PRIVATE - DO NOT USE! */
        UWORD pi_ymod;                  /* PRIVATE - DO NOT USE! */
        WORD pi_ety;                    /* PRIVATE - DO NOT USE! */
        UWORD pi_xpos;                  /* offset to start printing picture */
        UWORD pi_threshold;             /* threshold value (from prefs) */
        UWORD pi_tempwidth;             /* PRIVATE - DO NOT USE! */
        UWORD pi_flags;                 /* PRIVATE - DO NOT USE! */
};

#endif  /* DEVICES_PRTGFX_H */
```

```
#ifndef DEVICES_SCSIDISK_H
#define DEVICES_SCSIDISK_H
/*
**      $Filename: devices/scsidisk.h $
**      $Release: 2.04 $
**      $Revision: 36.2 $
**      $Date: 90/11/07 $
**
**      SCSI exec-level device command
**
**      (C) Copyright 1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/


#ifndef EXEC_TYPES_H
#include "exec/types.h"
#endif /* EXEC_TYPES_H */



/*------------------------------------------------------------------
 *
 *   SCSI Command
 *      Several Amiga SCSI controller manufacturers are converging on
 *      standard ways to talk to their controllers.  This include
 *      file describes an exec-device command (e.g. for hddisk.device)
 *      that can be used to issue SCSI commands
 *
 *   UNIT NUMBERS
 *      Unit numbers to the OpenDevice call have encoded in them which
 *      SCSI device is being referred to.  The three decimal digits of
 *      the unit number refer to the SCSI Target ID (bus address) in
 *      the 1's digit, the SCSI logical unit (LUN) in the 10's digit,
 *      and the controller board in the 100's digit.
 *
 *      Examples:
 *                0     drive at address 0
 *               12     LUN 1 on multiple drive controller at address 2
 *              104     second controller board, address 4
 *               88     not valid: both logical units and addresses
 *                      range from 0..7.
 *
 *   CAVEATS
 *      Original 2090 code did not support this command.
 *
 *      Commodore 2090/2090A unit numbers are different.  The SCSI
 *      logical unit is the 100's digit, and the SCSI Target ID
 *      is a permuted 1's digit: Target ID 0..6 maps to unit 3..9
 *      (7 is reserved for the controller).
 *
 *          Examples:
 *                3     drive at address 0
 *              109     drive at address 6, logical unit 1
 *                1     not valid: this is not a SCSI unit.  Perhaps
 *                      it's an ST506 unit.
 *
 *      Some controller boards generate a unique name (e.g. 2090A's
 *      iddisk.device) for the second controller board, instead of
 *      implementing the 100's digit.
 *
 *      There are optional restrictions on the alignment, bus
 *      accessability, and size of the data for the data phase.
 *      Be conservative to work with all manufacturer's controllers.
 *
 *------------------------------------------------------------------*/
```

```

#define HD_SCSICMD      28      /* issue a SCSI command to the unit */
                                /* io_Data points to a SCSICmd */
                                /* io_Length is sizeof(struct SCSICmd) */
                                /* io_Actual and io_Offset are not used */

struct SCSICmd {
    UWORD  *scsi_Data;          /* word aligned data for SCSI Data Phase */
                                /* (optional) data need not be byte aligned */
                                /* (optional) data need not be bus accessable */
    ULONG   scsi_Length;        /* even length of Data area */
                                /* (optional) data can have odd length */
                                /* (optional) data length can be > 2**24 */
    ULONG   scsi_Actual;        /* actual Data used */
    UBYTE  *scsi_Command;       /* SCSI Command (same options as scsi_Data) */
    UWORD   scsi_CmdLength;     /* length of Command */
    UWORD   scsi_CmdActual;     /* actual Command used */
    UBYTE   scsi_Flags;         /* includes intended data direction */
    UBYTE   scsi_Status;        /* SCSI status of command */
    UBYTE  *scsi_SenseData;     /* sense data: filled if SCSIF_[OLD]AUTOSENSE */
                                /* is set and scsi_Status has CHECK CONDITION */
                                /* (bit 1) set */
    UWORD   scsi_SenseLength;   /* size of scsi_SenseData, also bytes to */
                                /* request w/ SCSIF_AUTOSENSE, must be 4..255 */
    UWORD   scsi_SenseActual;   /* amount actually fetched (0 means no sense) */
};


/*----- scsi_Flags -----*/
#define SCSIF_WRITE             0       /* intended data direction is out */
#define SCSIF_READ              1       /* intended data direction is in */
#define SCSIB_READ_WRITE        0       /* (the bit to test) */

#define SCSIF_NOSENSE           0       /* no automatic request sense */
#define SCSIF_AUTOSENSE         2       /* do standard extended request sense */
                                        /* on check condition */
#define SCSIF_OLDAUTOSENSE      6       /* do 4 byte non-extended request */
                                        /* sense on check condition */
#define SCSIB_AUTOSENSE         1       /* (the bit to test) */
#define SCSIB_OLDAUTOSENSE      2       /* (the bit to test) */

/*----- SCSI io_Error values -----*/
#define HFERR_SelfUnit          40      /* cannot issue SCSI command to self */
#define HFERR_DMA               41      /* DMA error */
#define HFERR_Phase             42      /* illegal or unexpected SCSI phase */
#define HFERR_Parity            43      /* SCSI parity error */
#define HFERR_SelTimeout        44      /* Select timed out */
#define HFERR_BadStatus         45      /* status and/or sense error */

/*----- OpenDevice io_Error values -----*/
#define HFERR_NoBoard           50      /* Open failed for non-existant board */

#endif  /* DEVICES_SCSIDISK_H */
```

```
#ifndef DEVICES_SERIAL_H
#define DEVICES_SERIAL_H
/*
**      $Filename: devices/serial.h $
**      $Release: 2.04 $
**      $Revision: 33.6 $
**      $Date: 90/11/06 $
**
**      external declarations for the serial device
**
**      (C) Copyright 1985,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef   EXEC_IO_H
#include "exec/io.h"
#endif /* EXEC_IO_H */

                    /* array of termination char's */
                    /* to use,see serial.doc setparams */

 struct  IOTArray {
        ULONG TermArray0;
        ULONG TermArray1;
};


#define SER_DEFAULT_CTLCHAR 0x11130000  /* default chars for xON,xOFF */
/* You may change these via SETPARAMS.  At this time, parity is not
   calculated for xON/xOFF characters.  You must supply them with the
   desired parity. */

/*****************************************************************/
/* CAUTION !!  IF YOU ACCESS the serial.device, you MUST (!!!!) use an
   IOExtSer-sized structure or you may overlay innocent memory !! */
/*****************************************************************/

 struct  IOExtSer {
        struct   IOStdReq IOSer;

/*      STRUCT   MsgNode
*   0   APTR      Succ
*   4   APTR      Pred
*   8   UBYTE     Type
*   9   UBYTE     Pri
*   A   APTR      Name
*   E   APTR      ReplyPort
*  12   UWORD     MNLength
*      STRUCT   IOExt
*  14   APTR      io_Device
*  18   APTR      io_Unit
*  1C   UWORD     io_Command
*  1E   UBYTE     io_Flags
*  1F   UBYTE     io_Error
*      STRUCT   IOStdExt
*  20   ULONG     io_Actual
*  24   ULONG     io_Length
*  28   APTR      io_Data
*  2C   ULONG     io_Offset
*
*  30 */
   ULONG   io_CtlChar;    /* control char's (order = xON,xOFF,INQ,ACK) */
   ULONG   io_RBufLen;    /* length in bytes of serial port's read buffer */
   ULONG   io_ExtFlags;   /* additional serial flags (see bitdefs below) */
   ULONG   io_Baud;       /* baud rate requested (true baud) */
   ULONG   io_BrkTime;    /* duration of break signal in MICROseconds */
```

```
    struct  IOTArray io_TermArray; /* termination character array */
    UBYTE   io_ReadLen;    /* bits per read character (# of bits) */
    UBYTE   io_WriteLen;   /* bits per write character (# of bits) */
    UBYTE   io_StopBits;   /* stopbits for read (# of bits) */
    UBYTE   io_SerFlags;   /* see SerFlags bit definitions below  */
    UWORD   io_Status;
};
    /* status of serial port, as follows:
*                  BIT  ACTIVE  FUNCTION
*                   0    ---    reserved
*                   1    ---    reserved
*                   2   high    Connected to parallel "select" on the A1000.
*                               Connected to both the parallel "select" and
*                               serial "ring indicator" pins on the A500
*                               & A2000.  Take care when making cables.
*                   3   low     Data Set Ready
*                   4   low     Clear To Send
*                   5   low     Carrier Detect
*                   6   low     Ready To Send
*                   7   low     Data Terminal Ready
*                   8   high    read overrun
*                   9   high    break sent
*                  10   high    break received
*                  11   high    transmit x-OFFed
*                  12   high    receive x-OFFed
*                 13-15         reserved
*/

#define   SDCMD_QUERY           CMD_NONSTD      /* $09 */
#define   SDCMD_BREAK          (CMD_NONSTD+1)   /* $0A */
#define   SDCMD_SETPARAMS      (CMD_NONSTD+2)   /* $0B */


#define SERB_XDISABLED  7       /* io_SerFlags xOn-xOff feature disabled bit */
#define SERF_XDISABLED  (1<<7)  /*    "      xOn-xOff feature disabled mask */
#define SERB_EOFMODE    6       /*    "      EOF mode enabled bit */
#define SERF_EOFMODE    (1<<6)  /*    "      EOF mode enabled mask */
#define SERB_SHARED     5       /*    "      non-exclusive access bit */
#define SERF_SHARED     (1<<5)  /*    "      non-exclusive access mask */
#define SERB_RAD_BOOGIE 4       /*    "      high-speed mode active bit */
#define SERF_RAD_BOOGIE (1<<4)  /*    "      high-speed mode active mask */
#define SERB_QUEUEDBRK  3       /*    "      queue this Break ioRqst */
#define SERF_QUEUEDBRK  (1<<3)  /*    "      queue this Break ioRqst */
#define SERB_7WIRE      2       /*    "      RS232 7-wire protocol */
#define SERF_7WIRE      (1<<2)  /*    "      RS232 7-wire protocol */
#define SERB_PARTY_ODD  1       /*    "      parity feature enabled bit */
#define SERF_PARTY_ODD  (1<<1)  /*    "      parity feature enabled mask */
#define SERB_PARTY_ON   0       /*    "      parity-enabled bit */
#define SERF_PARTY_ON   (1<<0)  /*    "      parity-enabled mask */

/* These now refect the actual bit positions in the io_Status UWORD */
#define IO_STATB_XOFFREAD 12      /* io_Status receive currently xOFF'ed bit */
#define IO_STATF_XOFFREAD (1<<12)  /*    "     receive currently xOFF'ed mask */
#define IO_STATB_XOFFWRITE 11      /*    "     transmit currently xOFF'ed bit */
#define IO_STATF_XOFFWRITE (1<<11) /*    "     transmit currently xOFF'ed mask *
/
#define IO_STATB_READBREAK 10      /*    "     break was latest input bit */
#define IO_STATF_READBREAK (1<<10) /*    "     break was latest input mask */
#define IO_STATB_WROTEBREAK 9      /*    "     break was latest output bit */
#define IO_STATF_WROTEBREAK (1<<9) /*    "     break was latest output mask */
#define IO_STATB_OVERRUN 8         /*    "     status word RBF overrun bit */
#define IO_STATF_OVERRUN (1<<8)    /*    "     status word RBF overrun mask */


#define SEXTB_MSPON     1       /* io_ExtFlags. Use mark-space parity, */
                                /*          instead of odd-even. */
```

devices/serial.h

Page 3

```
#define SEXTF_MSPON     (1<<1)  /*   "    mark-space parity mask */
#define SEXTB_MARK      0       /*   "    if mark-space, use mark */
#define SEXTF_MARK      (1<<0)  /*   "    if mark-space, use mark mask */


#define SerErr_DevBusy          1
#define SerErr_BaudMismatch     2 /* baud rate not supported by hardware */
#define SerErr_BufErr           4 /* Failed to allocate new read buffer */
#define SerErr_InvParam         5
#define SerErr_LineErr          6
#define SerErr_ParityErr        9
#define SerErr_TimerErr         11 /*(See the serial/OpenDevice autodoc)*/
#define SerErr_BufOverflow      12
#define SerErr_NoDSR            13
#define SerErr_DetectedBreak    15


#ifdef DEVICES_SERIAL_H_OBSOLETE
#define SerErr_InvBaud          3       /* unused */
#define SerErr_NotOpen          7       /* unused */
#define SerErr_PortReset         8      /* unused */
#define SerErr_InitErr          10      /* unused */
#define SerErr_NoCTS            14      /* unused */

/* These defines refer to the HIGH ORDER byte of io_Status.  They have
   been replaced by the new, corrected ones above */
#define IOSTB_XOFFREAD  4       /* iost_hob receive currently xOFF'ed bit */
#define IOSTF_XOFFREAD  (1<<4)  /*    "     receive currently xOFF'ed mask */
#define IOSTB_XOFFWRITE 3       /*    "     transmit currently xOFF'ed bit */
#define IOSTF_XOFFWRITE (1<<3)  /*    "     transmit currently xOFF'ed mask */
#define IOSTB_READBREAK 2       /*    "     break was latest input bit */
#define IOSTF_READBREAK (1<<2)  /*    "     break was latest input mask */
#define IOSTB_WROTEBREAK 1      /*    "     break was latest output bit */
#define IOSTF_WROTEBREAK (1<<1) /*    "     break was latest output mask */
#define IOSTB_OVERRUN   0       /*    "     status word RBF overrun bit */
#define IOSTF_OVERRUN   (1<<0)  /*    "     status word RBF overrun mask */

#define IOSERB_BUFRREAD 7       /* io_Flags from read buffer bit */
#define IOSERF_BUFRREAD (1<<7)  /*    "     from read buffer mask */
#define IOSERB_QUEUED   6       /*    "     rqst-queued bit */
#define IOSERF_QUEUED   (1<<6)  /*    "     rqst-queued mask */
#define IOSERB_ABORT    5       /*    "     rqst-aborted bit */
#define IOSERF_ABORT    (1<<5)  /*    "     rqst-aborted mask */
#define IOSERB_ACTIVE   4       /*    "     rqst-qued-or-current bit */
#define IOSERF_ACTIVE   (1<<4)  /*    "     rqst-qued-or-current mask */
#endif


#define SERIALNAME     "serial.device"

#endif /* DEVICES_SERIAL_H */
```

devices/timer.h

Page 1

```
#ifndef DEVICES_TIMER_H
#define DEVICES_TIMER_H 1
/*
**      $Filename: devices/timer.h $
**      $Release: 2.04 $
**      $Revision: 36.16 $
**      $Date: 91/01/25 $
**
**      Timer device name and useful definitions.
**
**      (C) Copyright 1985,1985,1987,1988,1989,1990 Commodore-Amiga Inc.
**              All Rights Reserved
*/

#include <exec/types.h>
#include <exec/io.h>

/* unit defintions */
#define UNIT_MICROHZ    0
#define UNIT_VBLANK     1
#define UNIT_ECLOCK     2
#define UNIT_WAITUNTIL  3
#define UNIT_WAITECLOCK 4

#define TIMERNAME       "timer.device"

struct timeval {
    ULONG tv_secs;
    ULONG tv_micro;
};

struct EClockVal {
    ULONG ev_hi;
    ULONG ev_lo;
};

struct timerequest {
    struct IORequest tr_node;
    struct timeval tr_time;
};

/* IO_COMMAND to use for adding a timer */
#define TR_ADDREQUEST   CMD_NONSTD
#define TR_GETSYSTIME   (CMD_NONSTD+1)
#define TR_SETSYSTIME   (CMD_NONSTD+2)

#endif /* DEVICES_TIMER_H */
```

devices/trackdisk.h Page 1

```
#ifndef DEVICES_TRACKDISK_H
#define DEVICES_TRACKDISK_H

/*
**
**      $Filename: devices/trackdisk.h $
**      $Release: 2.04 $
**      $Revision: 33.13 $
**      $Date: 90/11/28 $
**
**      trackdisk device structure and value definitions
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**
*/

#ifndef EXEC_IO_H
#include "exec/io.h"
#endif

#ifndef EXEC_DEVICES_H
#include "exec/devices.h"
#endif

/*
 *--------------------------------------------------------------------
 *
 * Physical drive constants
 *
 *--------------------------------------------------------------------
 */

/* OBSOLETE -- use the TD_GETNUMTRACKS command! */
/*#define       NUMCYLS 80*/            /*  normal # of cylinders */
/*#define       MAXCYLS (NUMCYLS+20)*/  /* max # cyls to look for during cal */
/*#define       NUMHEADS 2*/
/*#define       NUMTRACKS (NUMCYLS*NUMHEADS)*/

#define NUMSECS 11
#define NUMUNITS 4

/*
 *--------------------------------------------------------------------
 *
 * Useful constants
 *
 *--------------------------------------------------------------------
 */

/*-- sizes before mfm encoding */
#define TD_SECTOR 512
#define TD_SECSHIFT 9           /* log TD_SECTOR */

/*
 *--------------------------------------------------------------------
 *
 * Driver Specific Commands
 *
 *--------------------------------------------------------------------
 */

/*
 *-- TD_NAME is a generic macro to get the name of the driver.  This
 *-- way if the name is ever changed you will pick up the change
 *-- automatically.
```

devices/trackdisk.h Page 2

```
 *--
 *-- Normal usage would be:
 *--
 *-- char internalName[] = TD_NAME;
 *--
 */

#define TD_NAME "trackdisk.device"

#define TDF_EXTCOM (1<<15)              /* for internal use only! */


#define TD_MOTOR        (CMD_NONSTD+0)  /* control the disk's motor */
#define TD_SEEK         (CMD_NONSTD+1)  /* explicit seek (for testing) */
#define TD_FORMAT       (CMD_NONSTD+2)  /* format disk */
#define TD_REMOVE       (CMD_NONSTD+3)  /* notify when disk changes */
#define TD_CHANGENUM    (CMD_NONSTD+4)  /* number of disk changes */
#define TD_CHANGESTATE  (CMD_NONSTD+5)  /* is there a disk in the drive? */
#define TD_PROTSTATUS   (CMD_NONSTD+6)  /* is the disk write protected? */
#define TD_RAWREAD      (CMD_NONSTD+7)  /* read raw bits from the disk */
#define TD_RAWWRITE     (CMD_NONSTD+8)  /* write raw bits to the disk */
#define TD_GETDRIVETYPE (CMD_NONSTD+9)  /* get the type of the disk drive */
#define TD_GETNUMTRACKS (CMD_NONSTD+10) /* # of tracks for this type drive */
#define TD_ADDCHANGEINT (CMD_NONSTD+11) /* TD_REMOVE done right */
#define TD_REMCHANGEINT (CMD_NONSTD+12) /* remove softint set by ADDCHANGEINT */
#define TD_GETGEOMETRY  (CMD_NONSTD+13) /* gets the disk geometry table */
#define TD_EJECT        (CMD_NONSTD+14) /* for those drives that support it */
#define TD_LASTCOMM     (CMD_NONSTD+15)

/*
 *
 * The disk driver has an "extended command" facility.  These commands
 * take a superset of the normal IO Request block.
 *
 */

#define ETD_WRITE       (CMD_WRITE|TDF_EXTCOM)
#define ETD_READ        (CMD_READ|TDF_EXTCOM)
#define ETD_MOTOR       (TD_MOTOR|TDF_EXTCOM)
#define ETD_SEEK        (TD_SEEK|TDF_EXTCOM)
#define ETD_FORMAT      (TD_FORMAT|TDF_EXTCOM)
#define ETD_UPDATE      (CMD_UPDATE|TDF_EXTCOM)
#define ETD_CLEAR       (CMD_CLEAR|TDF_EXTCOM)
#define ETD_RAWREAD     (TD_RAWREAD|TDF_EXTCOM)
#define ETD_RAWWRITE    (TD_RAWWRITE|TDF_EXTCOM)

/*
 *
 * extended IO has a larger than normal io request block.
 *
 */

struct IOExtTD {
        struct  IOStdReq iotd_Req;
        ULONG   iotd_Count;
        ULONG   iotd_SecLabel;
};

/*
 *  This is the structure returned by TD_DRIVEGEOMETRY
 *  Note that the layout can be defined three ways:
 *
 *  1. TotalSectors
 *  2. Cylinders and CylSectors
 *  3. Cylinders, Heads, and TrackSectors.
 *
 *  #1 is most accurate, #2 is less so, and #3 is least accurate.  All
 *  are usable, though #2 and #3 may waste some portion of the available
 *  space on some drives.
 */
struct DriveGeometry {
        ULONG   dg_SectorSize;          /* in bytes */
        ULONG   dg_TotalSectors;        /* total # of sectors on drive */
        ULONG   dg_Cylinders;           /* number of cylinders */
        ULONG   dg_CylSectors;          /* number of sectors/cylinder */
        ULONG   dg_Heads;               /* number of surfaces */
        ULONG   dg_TrackSectors;        /* number of sectors/track */
        ULONG   dg_BufMemType;          /* preferred buffer memory type */
                                        /* (usually MEMF_PUBLIC) */
        UBYTE   dg_DeviceType;          /* codes as defined in the SCSI-2 spec*/
        UBYTE   dg_Flags;               /* flags, including removable */
        UWORD   dg_Reserved;
};

/* device types */
#define DG_DIRECT_ACCESS        0
#define DG_SEQUENTIAL_ACCESS    1
#define DG_PRINTER              2
#define DG_PROCESSOR            3
#define DG_WORM                 4
#define DG_CDROM                5
#define DG_SCANNER              6
#define DG_OPTICAL_DISK         7
#define DG_MEDIUM_CHANGER       8
#define DG_COMMUNICATION        9
#define DG_UNKNOWN              31

/* flags */
#define DGB_REMOVABLE           0
#define DGF_REMOVABLE           1

/*
** raw read and write can be synced with the index pulse.  This flag
** in io request's IO_FLAGS field tells the driver that you want this.
*/

#define IOTDB_INDEXSYNC 4
#define IOTDF_INDEXSYNC (1<<4)
/*
** raw read and write can be synced with a $4489 sync pattern.  This flag
** in io request's IO_FLAGS field tells the driver that you want this.
*/
#define IOTDB_WORDSYNC  5
#define IOTDF_WORDSYNC (1<<5)


/* labels are TD_LABELSIZE bytes per sector */

#define TD_LABELSIZE 16

/*
** This is a bit in the FLAGS field of OpenDevice.  If it is set, then
** the driver will allow you to open all the disks that the trackdisk
** driver understands.  Otherwise only 3.5" disks will succeed.
*/

#define TDB_ALLOW_NON_3_5       0
#define TDF_ALLOW_NON_3_5       (1<<0)

/*
**  If you set the TDB_ALLOW_NON_3_5 bit in OpenDevice, then you don't
**  know what type of disk you really got.  These defines are for the
```

```
**  TD_GETDRIVETYPE command.  In addition, you can find out how many
**  tracks are supported via the TD_GETNUMTRACKS command.
*/

#define DRIVE3_5        1
#define DRIVE5_25       2
#define DRIVE3_5_150RPM 3

/*
 *--------------------------------------------------------------------
 *
 * Driver error defines
 *
 *--------------------------------------------------------------------
 */

#define TDERR_NotSpecified      20      /* general catchall */
#define TDERR_NoSecHdr          21      /* couldn't even find a sector */
#define TDERR_BadSecPreamble    22      /* sector looked wrong */
#define TDERR_BadSecID          23      /* ditto */
#define TDERR_BadHdrSum         24      /* header had incorrect checksum */
#define TDERR_BadSecSum         25      /* data had incorrect checksum */
#define TDERR_TooFewSecs        26      /* couldn't find enough sectors */
#define TDERR_BadSecHdr         27      /* another "sector looked wrong" */
#define TDERR_WriteProt         28      /* can't write to a protected disk */
#define TDERR_DiskChanged       29      /* no disk in the drive */
#define TDERR_SeekError         30      /* couldn't find track 0 */
#define TDERR_NoMem             31      /* ran out of memory */
#define TDERR_BadUnitNum        32      /* asked for a unit > NUMUNITS */
#define TDERR_BadDriveType      33      /* not a drive that trackdisk groks */
#define TDERR_DriveInUse        34      /* someone else allocated the drive */
#define TDERR_PostReset         35      /* user hit reset; awaiting doom */

/*
 *--------------------------------------------------------------------
 *
 * public portion of the unit structure
 *
 *--------------------------------------------------------------------
 */

struct TDU_PublicUnit {
        struct  Unit tdu_Unit;          /* base message port */
        UWORD   tdu_Comp01Track;        /* track for first precomp */
        UWORD   tdu_Comp10Track;        /* track for second precomp */
        UWORD   tdu_Comp11Track;        /* track for third precomp */
        ULONG   tdu_StepDelay;          /* time to wait after stepping */
        ULONG   tdu_SettleDelay;        /* time to wait after seeking */
        UBYTE   tdu_RetryCnt;           /* # of times to retry */
        UBYTE   tdu_PubFlags;           /* public flags, see below */
        UWORD   tdu_CurrTrk;            /* track the heads are over... */
                                        /* ONLY ACCESS WHILE UNIT IS STOPPED! */
        ULONG   tdu_CalibrateDelay;     /* time to wait after stepping */
                                        /* during a recalibrate */
        ULONG   tdu_Counter;            /* counter for disk changes... */
                                        /* ONLY ACCESS WHILE UNIT IS STOPPED! */
};

/* flags for tdu_PubFlags */
#define TDPB_NOCLICK    0
#define TDPF_NOCLICK    (1L << 0)

#endif  /* DEVICES_TRACKDISK_H */
```

dos/datetime.h

```
#ifndef DOS_DATETIME_H
#define DOS_DATETIME_H

/*
**      $Filename: dos/datetime.h $
**      $Release: 2.04 $
**      $Revision: 36.7 $
**      $Date: 90/07/12 $
**
**      Date and time C header for AmigaDOS
**
**      (C) Copyright 1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**
*/

#ifndef DOS_DOS_H
#include "dos/dos.h"
#endif

/*
 *      Data structures and equates used by the V1.4 DOS functions
 * StrtoDate() and DatetoStr()
 */

/*--------- String/Date structures etc */
struct DateTime {
        struct DateStamp dat_Stamp;     /* DOS DateStamp */
        UBYTE   dat_Format;             /* controls appearance of dat_StrDate */
        UBYTE   dat_Flags;              /* see BITDEF's below */
        UBYTE   *dat_StrDay;            /* day of the week string */
        UBYTE   *dat_StrDate;           /* date string */
        UBYTE   *dat_StrTime;           /* time string */
};

/* You need this much room for each of the DateTime strings: */
#define LEN_DATSTRING   16

/*      flags for dat_Flags */

#define DTB_SUBST       0               /* substitute Today, Tomorrow, etc. */
#define DTF_SUBST       1
#define DTB_FUTURE      1               /* day of the week is in future */
#define DTF_FUTURE      2

/*
 *      date format values
 */

#define FORMAT_DOS      0               /* dd-mmm-yy */
#define FORMAT_INT      1               /* yy-mm-dd  */
#define FORMAT_USA      2               /* mm-dd-yy  */
#define FORMAT_CDN      3               /* dd-mm-yy  */
#define FORMAT_MAX      FORMAT_CDN

#endif /* DOS_DATETIME_H */
```

dos/dos.h

```
#ifndef DOS_DOS_H
#define DOS_DOS_H
/*
**      $Filename: dos/dos.h $
**      $Release: 2.04 $
**      $Revision: 36.20 $
**      $Date: 91/02/13 $
**
**      Standard C header for AmigaDOS
**
**      (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_TYPES_H
#include "exec/types.h"
#endif


#define  DOSNAME  "dos.library"

/* Predefined Amiga DOS global constants */

#define DOSTRUE (-1L)
#define DOSFALSE (0L)

/* Mode parameter to Open() */
#define MODE_OLDFILE         1005   /* Open existing file read/write
                                     * positioned at beginning of file. */
#define MODE_NEWFILE         1006   /* Open freshly created file (delete
                                     * old file) read/write, exclusive lock. */
#define MODE_READWRITE       1004   /* Open old file w/shared lock,
                                     * creates file if doesn't exist. */

/* Relative position to Seek() */
#define OFFSET_BEGINNING    -1      /* relative to Begining Of File */
#define OFFSET_CURRENT       0      /* relative to Current file position */
#define OFFSET_END           1      /* relative to End Of File    */

#define OFFSET_BEGINING     OFFSET_BEGINNING  /* ancient compatibility */

#define BITSPERBYTE          8
#define BYTESPERLONG         4
#define BITSPERLONG          32
#define MAXINT               0x7FFFFFFF
#define MININT               0x80000000

/* Passed as type to Lock() */
#define SHARED_LOCK          -2     /* File is readable by others */
#define ACCESS_READ          -2     /* Synonym */
#define EXCLUSIVE_LOCK       -1     /* No other access allowed    */
#define ACCESS_WRITE         -1     /* Synonym */

struct DateStamp {
   LONG  ds_Days;                 /* Number of days since Jan. 1, 1978 */
   LONG  ds_Minute;               /* Number of minutes past midnight */
   LONG  ds_Tick;                 /* Number of ticks past minute */
}; /* DateStamp */

#define TICKS_PER_SECOND      50   /* Number of ticks in one second */

/* Returned by Examine() and ExNext(), must be on a 4 byte boundary */
struct FileInfoBlock {
   LONG   fib_DiskKey;
   LONG   fib_DirEntryType;  /* Type of Directory. If < 0, then a plain file.
                              * If > 0 a directory */
   char   fib_FileName[108]; /* Null terminated. Max 30 chars used for now */
   LONG   fib_Protection;    /* bit mask of protection, rwxd are 3-0.      */
   LONG   fib_EntryType;
   LONG   fib_Size;          /* Number of bytes in file */
   LONG   fib_NumBlocks;     /* Number of blocks in file */
   struct DateStamp fib_Date;/* Date file last changed */
   char   fib_Comment[80];  /* Null terminated comment associated with file */
   char   fib_Reserved[36];
}; /* FileInfoBlock */

/* FIB stands for FileInfoBlock */

/* FIBB are bit definitions, FIBF are field definitions */
#define FIBB_SCRIPT    6        /* program is a script (execute) file */
#define FIBB_PURE      5        /* program is reentrant and rexecutable */
#define FIBB_ARCHIVE   4        /* cleared whenever file is changed */
#define FIBB_READ      3        /* ignored by old filesystem */
#define FIBB_WRITE     2        /* ignored by old filesystem */
#define FIBB_EXECUTE   1        /* ignored by system, used by Shell */
#define FIBB_DELETE    0        /* prevent file from being deleted */
#define FIBF_SCRIPT    (1<<FIBB_SCRIPT)
#define FIBF_PURE      (1<<FIBB_PURE)
#define FIBF_ARCHIVE   (1<<FIBB_ARCHIVE)
#define FIBF_READ      (1<<FIBB_READ)
#define FIBF_WRITE     (1<<FIBB_WRITE)
#define FIBF_EXECUTE   (1<<FIBB_EXECUTE)
#define FIBF_DELETE    (1<<FIBB_DELETE)

/* Standard maximum length for an error string from fault.  However, most */
/* error strings should be kept under 60 characters if possible.  Don't   */
/* forget space for the header you pass in. */
#define FAULT_MAX       82

/* All BCPL data must be long word aligned.  BCPL pointers are the long word
 *  address (i.e byte address divided by 4 (>>2)) */
typedef long  BPTR;                 /* Long word pointer */
typedef long  BSTR;                 /* Long word pointer to BCPL string  */

/* Convert BPTR to typical C pointer */
#ifdef OBSOLETE_LIBRARIES_DOS_H
#define BADDR( bptr )     (((ULONG)bptr) << 2)
#else
/* This one has no problems with CASTing */
#define BADDR(x)          ((APTR)((ULONG)(x) << 2))
#endif
/* Convert address into a BPTR */
#define MKBADDR(x)        (((LONG)(x)) >> 2)

/* BCPL strings have a length in the first byte and then the characters.
 * For example:  s[0]=3 s[1]=S s[2]=Y s[3]=S                             */

/* returned by Info(), must be on a 4 byte boundary */
struct InfoData {
   LONG   id_NumSoftErrors;     /* number of soft errors on disk */
   LONG   id_UnitNumber;        /* Which unit disk is (was) mounted on */
   LONG   id_DiskState;         /* See defines below */
   LONG   id_NumBlocks;         /* Number of blocks on disk */
   LONG   id_NumBlocksUsed;     /* Number of block in use */
   LONG   id_BytesPerBlock;
   LONG   id_DiskType;          /* Disk Type code */
   BPTR   id_VolumeNode;        /* BCPL pointer to volume node */
   LONG   id_InUse;             /* Flag, zero if not in use */
}; /* InfoData */

/* ID stands for InfoData */
        /* Disk states */
```

```
#define ID_WRITE_PROTECTED 80    /* Disk is write protected */
#define ID_VALIDATING      81    /* Disk is currently being validated */
#define ID_VALIDATED       82    /* Disk is consistent and writeable */

        /* Disk types */
#define ID_NO_DISK_PRESENT      (-1)
#define ID_UNREADABLE_DISK      (0x42414400L)   /* 'BAD\0' */
#define ID_DOS_DISK             (0x444F5300L)   /* 'DOS\0' */
#define ID_FFS_DISK             (0x444F5301L)   /* 'DOS\1' */
#define ID_NOT_REALLY_DOS       (0x4E444F53L)   /* 'NDOS'  */
#define ID_KICKSTART_DISK       (0x4B49434BL)   /* 'KICK'  */
#define ID_MSDOS_DISK           (0x4d534400L)   /* 'MSD\0' */

/* Errors from IoErr(), etc. */
#define ERROR_NO_FREE_STORE               103
#define ERROR_TASK_TABLE_FULL             105
#define ERROR_BAD_TEMPLATE                114
#define ERROR_BAD_NUMBER                  115
#define ERROR_REQUIRED_ARG_MISSING        116
#define ERROR_KEY_NEEDS_ARG               117
#define ERROR_TOO_MANY_ARGS               118
#define ERROR_UNMATCHED_QUOTES            119
#define ERROR_LINE_TOO_LONG               120
#define ERROR_FILE_NOT_OBJECT             121
#define ERROR_INVALID_RESIDENT_LIBRARY    122
#define ERROR_NO_DEFAULT_DIR              201
#define ERROR_OBJECT_IN_USE               202
#define ERROR_OBJECT_EXISTS               203
#define ERROR_DIR_NOT_FOUND               204
#define ERROR_OBJECT_NOT_FOUND            205
#define ERROR_BAD_STREAM_NAME             206
#define ERROR_OBJECT_TOO_LARGE            207
#define ERROR_ACTION_NOT_KNOWN            209
#define ERROR_INVALID_COMPONENT_NAME      210
#define ERROR_INVALID_LOCK                211
#define ERROR_OBJECT_WRONG_TYPE           212
#define ERROR_DISK_NOT_VALIDATED          213
#define ERROR_DISK_WRITE_PROTECTED        214
#define ERROR_RENAME_ACROSS_DEVICES       215
#define ERROR_DIRECTORY_NOT_EMPTY         216
#define ERROR_TOO_MANY_LEVELS             217
#define ERROR_DEVICE_NOT_MOUNTED          218
#define ERROR_SEEK_ERROR                  219
#define ERROR_COMMENT_TOO_BIG             220
#define ERROR_DISK_FULL                   221
#define ERROR_DELETE_PROTECTED            222
#define ERROR_WRITE_PROTECTED             223
#define ERROR_READ_PROTECTED              224
#define ERROR_NOT_A_DOS_DISK              225
#define ERROR_NO_DISK                     226
#define ERROR_NO_MORE_ENTRIES             232
/* added for 1.4 */
#define ERROR_IS_SOFT_LINK                233
#define ERROR_OBJECT_LINKED               234
#define ERROR_BAD_HUNK                    235
#define ERROR_NOT_IMPLEMENTED             236
#define ERROR_RECORD_NOT_LOCKED           240
#define ERROR_LOCK_COLLISION              241
#define ERROR_LOCK_TIMEOUT                242
#define ERROR_UNLOCK_ERROR                243

/* error codes 303-305 are defined in dosasl.h */

/* These are the return codes used by convention by AmigaDOS commands */
/* See FAILAT and IF for relvance to EXECUTE files                    */
#define RETURN_OK                           0  /* No problems, success */
```

dos/dos.h Page 4

```
#define RETURN_WARN                        5  /* A warning only */
#define RETURN_ERROR                      10  /* Something wrong */
#define RETURN_FAIL                       20  /* Complete or severe failure*/

/* Bit numbers that signal you that a user has issued a break */
#define SIGBREAKB_CTRL_C   12
#define SIGBREAKB_CTRL_D   13
#define SIGBREAKB_CTRL_E   14
#define SIGBREAKB_CTRL_F   15

/* Bit fields that signal you that a user has issued a break */
/* for example:  if (SetSignal(0,0) & SIGBREAKF_CTRL_C) cleanup_and_exit(); */
#define SIGBREAKF_CTRL_C   (1<<SIGBREAKB_CTRL_C)
#define SIGBREAKF_CTRL_D   (1<<SIGBREAKB_CTRL_D)
#define SIGBREAKF_CTRL_E   (1<<SIGBREAKB_CTRL_E)
#define SIGBREAKF_CTRL_F   ((long)1<<SIGBREAKB_CTRL_F)

/* Values returned by SameLock() */
#define LOCK_SAME               0
#define LOCK_SAME_HANDLER       1       /* actually same volume */
#define LOCK_DIFFERENT          -1

/* types for ChangeMode() */
#define CHANGE_LOCK     0
#define CHANGE_FH       1

/* Values for MakeLink() */
#define LINK_HARD       0
#define LINK_SOFT       1       /* softlinks are not fully supported yet */

/* values returned by ReadItem */
#define ITEM_EQUAL      -2              /* "=" Symbol */
#define ITEM_ERROR      -1              /* error */
#define ITEM_NOTHING    0               /* *N, ;, endstreamch */
#define ITEM_UNQUOTED   1               /* unquoted item */
#define ITEM_QUOTED     2               /* quoted item */

/* types for AllocDosObject/FreeDosObject */
#define DOS_FILEHANDLE          0       /* few people should use this */
#define DOS_EXALLCONTROL        1       /* Must be used to allocate this! */
#define DOS_FIB                 2       /* useful */
#define DOS_STDPKT              3       /* for doing packet-level I/O */
#define DOS_CLI                 4       /* for shell-writers, etc */
#define DOS_RDARGS              5       /* for ReadArgs if you pass it in */

#endif  /* DOS_DOS_H */
```

dos/dosasl.h Page 1

```
#ifndef DOS_DOSASL_H
#define DOS_DOSASL_H
/*
**
**      $Filename: dos/dosasl.h $
**      $Release: 2.04 $
**      $Revision: 36.15 $
**      $Date: 91/01/27 $
**
**      Pattern-matching structure definitions
**
**      (C) Copyright 1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**
*/

#ifndef EXEC_LIBRARIES_H
#include <exec/libraries.h>
#endif

#ifndef EXEC_LISTS_H
#include <exec/lists.h>
#endif

#ifndef DOS_DOS_H
#include <dos/dos.h>
#endif


/***********************************************************************
************************ PATTERN MATCHING ******************************
************************************************************************

* structure expected by MatchFirst, MatchNext.
* Allocate this structure and initialize it as follows:
*
* Set ap_BreakBits to the signal bits (CDEF) that you want to take a
* break on, or NULL, if you don't want to convenience the user.
*
* If you want to have the FULL PATH NAME of the files you found,
* allocate a buffer at the END of this structure, and put the size of
* it into ap_Strlen.  If you don't want the full path name, make sure
* you set ap_Strlen to zero.  In this case, the name of the file, and stats
* are available in the ap_Info, as per usual.
*
* Then call MatchFirst() and then afterwards, MatchNext() with this structure.
* You should check the return value each time (see below) and take the
* appropriate action, ultimately calling MatchEnd() when there are
* no more files and you are done.  You can tell when you are done by
* checking for the normal AmigaDOS return code ERROR_NO_MORE_ENTRIES.
*
*/

struct AnchorPath {
        struct AChain   *ap_Base;       /* pointer to first anchor */
#define ap_First ap_Base
        struct AChain   *ap_Last;       /* pointer to last anchor */
#define ap_Current ap_Last
        LONG    ap_BreakBits;   /* Bits we want to break on */
        LONG    ap_FoundBreak;  /* Bits we broke on. Also returns ERROR_BREAK */
        BYTE    ap_Flags;       /* New use for extra word. */
        BYTE    ap_Reserved;
        WORD    ap_Strlen;      /* This is what ap_Length used to be */
#define ap_Length ap_Flags      /* Old compatability for LONGWORD ap_Length */
        struct  FileInfoBlock ap_Info;
        UBYTE   ap_Buf[1];      /* Buffer for path name, allocated by user */
        /* FIX! */
};


#define APB_DOWILD      0       /* User option ALL */
#define APF_DOWILD      1

#define APB_ITSWILD     1       /* Set by MatchFirst, used by MatchNext  */
#define APF_ITSWILD     2       /* Application can test APB_ITSWILD, too */
                                /* (means that there's a wildcard        */
                                /* in the pattern after calling          */
                                /* MatchFirst).                          */

#define APB_DODIR       2       /* Bit is SET if a DIR node should be */
#define APF_DODIR       4       /* entered. Application can RESET this */
                                /* bit after MatchFirst/MatchNext to AVOID */
                                /* entering a dir. */

#define APB_DIDDIR      3       /* Bit is SET for an "expired" dir node. */
#define APF_DIDDIR      8

#define APB_NOMEMERR    4       /* Set on memory error */
#define APF_NOMEMERR    16

#define APB_DODOT       5       /* If set, allow conversion of '.' to */
#define APF_DODOT       32      /* CurrentDir */

#define APB_DirChanged  6       /* ap_Current->an_Lock changed */
#define APF_DirChanged  64      /* since last MatchNext call */


struct AChain {
        struct AChain *an_Child;
        struct AChain *an_Parent;
        BPTR    an_Lock;
        struct FileInfoBlock an_Info;
        BYTE    an_Flags;
        UBYTE   an_String[1];   /* FIX!! */
};

#define DDB_PatternBit  0
#define DDF_PatternBit  1
#define DDB_ExaminedBit 1
#define DDF_ExaminedBit 2
#define DDB_Completed   2
#define DDF_Completed   4
#define DDB_AllBit      3
#define DDF_AllBit      8
#define DDB_Single      4
#define DDF_Single      16

/*
 * Constants used by wildcard routines, these are the pre-parsed tokens
 * referred to by pattern match.  It is not necessary for you to do
 * anything about these, MatchFirst() MatchNext() handle all these for you.
 */

#define P_ANY           0x80    /* Token for '*' or '#?  */
#define P_SINGLE        0x81    /* Token for '?' */
#define P_ORSTART       0x82    /* Token for '(' */
#define P_ORNEXT        0x83    /* Token for '|' */
#define P_OREND 0x84    /* Token for ')' */
#define P_NOT           0x85    /* Token for '~' */
#define P_NOTEND        0x86    /* Token for */
#define P_NOTCLASS      0x87    /* Token for '^' */
#define P_CLASS 0x88    /* Token for '[]' */
```

```
#define P_REPBEG        0x89    /* Token for '[' */
#define P_REPEND        0x8A    /* Token for ']' */
#define P_STOP          0x8B    /* token to force end of evaluation */

/* Values for an_Status, NOTE: These are the actual bit numbers */

#define COMPLEX_BIT     1       /* Parsing complex pattern */
#define EXAMINE_BIT     2       /* Searching directory */

/*
 * Returns from MatchFirst(), MatchNext()
 * You can also get dos error returns, such as ERROR_NO_MORE_ENTRIES,
 * these are in the dos.h file.
 */

#define ERROR_BUFFER_OVERFLOW   303     /* User or internal buffer overflow */
#define ERROR_BREAK             304     /* A break character was received */
#define ERROR_NOT_EXECUTABLE    305     /* A file has E bit cleared */

#endif /* DOS_DOSASL_H */
```

dos/dosextens.h

Page 1

```
#ifndef DOS_DOSEXTENS_H
#define DOS_DOSEXTENS_H
/*
**      $Filename: dos/dosextens.h $
**      $Release: 2.04 $
**      $Revision: 36.35 $
**      $Date: 91/02/25 $
**
**      DOS structures not needed for the casual AmigaDOS user
**
**      (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_TASKS_H
#include "exec/tasks.h"
#endif
#ifndef EXEC_PORTS_H
#include "exec/ports.h"
#endif
#ifndef EXEC_LIBRARIES_H
#include "exec/libraries.h"
#endif
#ifndef EXEC_SEMAPHORES_H
#include "exec/semaphores.h"
#endif
#ifndef DEVICES_TIMER_H
#include "devices/timer.h"
#endif

#ifndef DOS_DOS_H
#include "dos/dos.h"
#endif

/* All DOS processes have this structure */
/* Create and Device Proc returns pointer to the MsgPort in this structure */
/* dev_proc = (struct Process *) (DeviceProc(..) - sizeof(struct Task)); */

struct Process {
    struct  Task    pr_Task;
    struct  MsgPort pr_MsgPort; /* This is BPTR address from DOS functions  */
    WORD    pr_Pad;             /* Remaining variables on 4 byte boundaries */
    BPTR    pr_SegList;         /* Array of seg lists used by this process  */
    LONG    pr_StackSize;       /* Size of process stack in bytes           */
    APTR    pr_GlobVec;         /* Global vector for this process (BCPL)    */
    LONG    pr_TaskNum;         /* CLI task number of zero if not a CLI     */
    BPTR    pr_StackBase;       /* Ptr to high memory end of process stack  */
    LONG    pr_Result2;         /* Value of secondary result from last call */
    BPTR    pr_CurrentDir;      /* Lock associated with current directory   */
    BPTR    pr_CIS;             /* Current CLI Input Stream                 */
    BPTR    pr_COS;             /* Current CLI Output Stream                */
    APTR    pr_ConsoleTask;     /* Console handler process for current window*/
    APTR    pr_FileSystemTask;  /* File handler process for current drive   */
    BPTR    pr_CLI;             /* pointer to CommandLineInterface          */
    APTR    pr_ReturnAddr;      /* pointer to previous stack frame          */
    APTR    pr_PktWait;         /* Function to be called when awaiting msg  */
    APTR    pr_WindowPtr;       /* Window for error printing                */

    /* following definitions are new with 2.0 */
    BPTR    pr_HomeDir;         /* Home directory of executing program      */
    LONG    pr_Flags;           /* flags telling dos about process          */
    void    (*pr_ExitCode)();   /* code to call on exit of program or NULL  */
    LONG    pr_ExitData;        /* Passed as an argument to pr_ExitCode.    */
    UBYTE   *pr_Arguments;      /* Arguments passed to the process at start */
    struct MinList pr_LocalVars; /* Local environment variables             */
    ULONG   pr_ShellPrivate;    /* for the use of the current shell         */
```

dos/dosextens.h

Page 2

```
    BPTR    pr_CES;             /* Error stream - if NULL, use pr_COS       */
};  /* Process */

/*
 * Flags for pr_Flags
 */
#define PRB_FREESEGLIST         0
#define PRF_FREESEGLIST         1
#define PRB_FREECURRDIR         1
#define PRF_FREECURRDIR         2
#define PRB_FREECLI             2
#define PRF_FREECLI             4
#define PRB_CLOSEINPUT          3
#define PRF_CLOSEINPUT          8
#define PRB_CLOSEOUTPUT         4
#define PRF_CLOSEOUTPUT         16
#define PRB_FREEARGS            5
#define PRF_FREEARGS            32

/* The long word address (BPTR) of this structure is returned by
 * Open() and other routines that return a file.  You need only worry
 * about this struct to do async io's via PutMsg() instead of
 * standard file system calls */

struct FileHandle {
   struct Message *fh_Link;      /* EXEC message              */
   struct MsgPort *fh_Port;      /* Reply port for the packet */
   struct MsgPort *fh_Type;      /* Port to do PutMsg() to
                                  * Address is negative if a plain file */
   LONG fh_Buf;
   LONG fh_Pos;
   LONG fh_End;
   LONG fh_Funcs;
#define fh_Func1 fh_Funcs
   LONG fh_Func2;
   LONG fh_Func3;
   LONG fh_Args;
#define fh_Arg1 fh_Args
   LONG fh_Arg2;
}; /* FileHandle */

/* This is the extension to EXEC Messages used by DOS */

struct DosPacket {
   struct Message *dp_Link;      /* EXEC message              */
   struct MsgPort *dp_Port;      /* Reply port for the packet */
                                 /* Must be filled in each send. */
   LONG dp_Type;                 /* See ACTION_... below and
                                  * 'R' means Read, 'W' means Write to the
                                  * file system */
   LONG dp_Res1;                 /* For file system calls this is the result
                                  * that would have been returned by the
                                  * function, e.g. Write ('W') returns actual
                                  * length written */
   LONG dp_Res2;                 /* For file system calls this is what would
                                  * have been returned by IoErr() */
/*  Device packets common equivalents */
#define dp_Action  dp_Type
#define dp_Status  dp_Res1
#define dp_Status2 dp_Res2
#define dp_BufAddr dp_Arg1
   LONG dp_Arg1;
   LONG dp_Arg2;
   LONG dp_Arg3;
   LONG dp_Arg4;
   LONG dp_Arg5;
   LONG dp_Arg6;
   LONG dp_Arg7;
}; /* DosPacket */

/* A Packet does not require the Message to be before it in memory, but
 * for convenience it is useful to associate the two.
 * Also see the function init_std_pkt for initializing this structure */

struct StandardPacket {
   struct Message   sp_Msg;
   struct DosPacket sp_Pkt;
}; /* StandardPacket */

/* Packet types */
#define ACTION_NIL              0
#define ACTION_STARTUP          0
#define ACTION_GET_BLOCK        2       /* OBSOLETE */
#define ACTION_SET_MAP          4
#define ACTION_DIE              5
#define ACTION_EVENT            6
#define ACTION_CURRENT_VOLUME   7
#define ACTION_LOCATE_OBJECT    8
#define ACTION_RENAME_DISK      9
#define ACTION_WRITE            'W'
#define ACTION_READ             'R'
#define ACTION_FREE_LOCK        15
#define ACTION_DELETE_OBJECT    16
#define ACTION_RENAME_OBJECT    17
#define ACTION_MORE_CACHE       18
#define ACTION_COPY_DIR         19
#define ACTION_WAIT_CHAR        20
#define ACTION_SET_PROTECT      21
#define ACTION_CREATE_DIR       22
#define ACTION_EXAMINE_OBJECT   23
#define ACTION_EXAMINE_NEXT     24
#define ACTION_DISK_INFO        25
#define ACTION_INFO             26
#define ACTION_FLUSH            27
#define ACTION_SET_COMMENT      28
#define ACTION_PARENT           29
#define ACTION_TIMER            30
#define ACTION_INHIBIT          31
#define ACTION_DISK_TYPE        32
#define ACTION_DISK_CHANGE      33
#define ACTION_SET_DATE         34

#define ACTION_SCREEN_MODE      994

#define ACTION_READ_RETURN      1001
#define ACTION_WRITE_RETURN     1002
#define ACTION_SEEK             1008
#define ACTION_FINDUPDATE       1004
#define ACTION_FINDINPUT        1005
#define ACTION_FINDOUTPUT       1006
#define ACTION_END              1007
#define ACTION_SET_FILE_SIZE    1022    /* fast file system only in 1.3 */
#define ACTION_WRITE_PROTECT    1023    /* fast file system only in 1.3 */

/* new 2.0 packets */
#define ACTION_SAME_LOCK        40
#define ACTION_CHANGE_SIGNAL    995
#define ACTION_FORMAT           1020
#define ACTION_MAKE_LINK        1021
/**/
/**/
#define ACTION_READ_LINK        1024
```

```
#define ACTION_FH_FROM_LOCK     1026
#define ACTION_IS_FILESYSTEM    1027
#define ACTION_CHANGE_MODE      1028
/**/
#define ACTION_COPY_DIR_FH      1030
#define ACTION_PARENT_FH        1031
#define ACTION_EXAMINE_ALL      1033
#define ACTION_EXAMINE_FH       1034

#define ACTION_LOCK_RECORD      2008
#define ACTION_FREE_RECORD      2009

#define ACTION_ADD_NOTIFY       4097
#define ACTION_REMOVE_NOTIFY    4098

/*
 * A structure for holding error messages - stored as array with error == 0
 * for the last entry.
 */
struct ErrorString {
        LONG  *estr_Nums;
        UBYTE *estr_Strings;
};

/* DOS library node structure.
 * This is the data at positive offsets from the library node.
 * Negative offsets from the node is the jump table to DOS functions
 * node = (struct DosLibrary *) OpenLibrary( "dos.library" .. )      */

struct DosLibrary {
    struct Library dl_lib;
    struct RootNode *dl_Root; /* Pointer to RootNode, described below */
    APTR    dl_GV;            /* Pointer to BCPL global vector        */
    LONG    dl_A2;            /* Private register dump of DOS         */
    LONG    dl_A5;
    LONG    dl_A6;
    struct ErrorString *dl_Errors;  /* pointer to array of error msgs */
    struct timerequest *dl_TimeReq; /* private pointer to timer request */
    struct Library     *dl_UtilityBase; /* private ptr to utility library */
};  /*  DosLibrary */

/*                            */

struct RootNode {
    BPTR    rn_TaskArray;           /* [0] is max number of CLI's
                                     * [1] is APTR to process id of CLI 1
                                     * [n] is APTR to process id of CLI n */
    BPTR    rn_ConsoleSegment; /* SegList for the CLI                        */
    struct  DateStamp rn_Time; /* Current time                               */
    LONG    rn_RestartSeg;     /* SegList for the disk validator process     */
    BPTR    rn_Info;           /* Pointer to the Info structure              */
    BPTR    rn_FileHandlerSegment; /* segment for a file handler             */
    struct MinList rn_CliList; /* new list of all CLI processes */
                               /* the first cpl_Array is also rn_TaskArray */
    struct MsgPort *rn_BootProc; /* private ptr to msgport of boot fs        */
    BPTR    rn_ShellSegment;   /* seglist for Shell (for NewShell)           */
    LONG    rn_Flags;          /* dos flags */
};  /* RootNode */

#define RNB_WILDSTAR    24
#define RNF_WILDSTAR    (1L<<24)
#define RNB_PRIVATE1    1       /* private for dos */
#define RNF_PRIVATE1    2

/* ONLY to be allocated by DOS! */
struct CliProcList {
        struct MinNode cpl_Node;
        LONG cpl_First;     /* number of first entry in array */
        struct MsgPort **cpl_Array;
                             /* [0] is max number of CLI's in this entry (n)
                              * [1] is CPTR to process id of CLI cpl_First
                              * [n] is CPTR to process id of CLI cpl_First+n-1
                              */
};

struct DosInfo {
    BPTR    di_McName;         /* PRIVATE: system resident module list      */
#define di_ResList di_McName
    BPTR    di_DevInfo;        /* Device List                               */
    BPTR    di_Devices;        /* Currently zero                            */
    BPTR    di_Handlers;       /* Currently zero                            */
    APTR    di_NetHand;        /* Network handler processid; currently zero */
    struct  SignalSemaphore di_DevLock;    /* do NOT access directly! */
    struct  SignalSemaphore di_EntryLock;  /* do NOT access directly! */
    struct  SignalSemaphore di_DeleteLock; /* do NOT access directly! */
};  /* DosInfo */

/* structure for the Dos resident list.  Do NOT allocate these, use       */
/* AddSegment(), and heed the warnings in the autodocs!                   */

struct Segment {
        BPTR seg_Next;
        LONG seg_UC;
        BPTR seg_Seg;
        UBYTE seg_Name[4];      /* actually the first 4 chars of BSTR name */
};

#define CMD_SYSTEM      -1
#define CMD_INTERNAL    -2
#define CMD_DISABLED    -999


/* DOS Processes started from the CLI via RUN or NEWCLI have this additional
 * set to data associated with them */

struct CommandLineInterface {
    LONG   cli_Result2;        /* Value of IoErr from last command        */
    BSTR   cli_SetName;        /* Name of current directory               */
    BPTR   cli_CommandDir;     /* Head of the path locklist               */
    LONG   cli_ReturnCode;     /* Return code from last command           */
    BSTR   cli_CommandName;    /* Name of current command                 */
    LONG   cli_FailLevel;      /* Fail level (set by FAILAT)              */
    BSTR   cli_Prompt;         /* Current prompt (set by PROMPT)          */
    BPTR   cli_StandardInput;  /* Default (terminal) CLI input            */
    BPTR   cli_CurrentInput;   /* Current CLI input                       */
    BSTR   cli_CommandFile;    /* Name of EXECUTE command file            */
    LONG   cli_Interactive;    /* Boolean; True if prompts required       */
    LONG   cli_Background;     /* Boolean; True if CLI created by RUN     */
    BPTR   cli_CurrentOutput;  /* Current CLI output                      */
    LONG   cli_DefaultStack;   /* Stack size to be obtained in long words */
    BPTR   cli_StandardOutput; /* Default (terminal) CLI output           */
    BPTR   cli_Module;         /* SegList of currently loaded command     */
};  /* CommandLineInterface */

/* This structure can take on different values depending on whether it is
 * a device, an assigned directory, or a volume.  Below is the structure
 * reflecting volumes only.  Following that is the structure representing
 * only devices. Following that is the unioned structure representing all
 * the values
 */

/* structure representing a volume */
```

```

struct DeviceList {
    BPTR                dl_Next;        /* bptr to next device list */
    LONG                dl_Type;        /* see DLT below */
    struct MsgPort *    dl_Task;        /* ptr to handler task */
    BPTR                dl_Lock;        /* not for volumes */
    struct DateStamp    dl_VolumeDate;  /* creation date */
    BPTR                dl_LockList;    /* outstanding locks */
    LONG                dl_DiskType;    /* 'DOS', etc */
    LONG                dl_unused;
    BSTR                dl_Name;        /* bptr to bcpl name */
};

/* device structure (same as the DeviceNode structure in filehandler.h) */

struct        DevInfo {
    BPTR  dvi_Next;
    LONG  dvi_Type;
    APTR  dvi_Task;
    BPTR  dvi_Lock;
    BSTR  dvi_Handler;
    LONG  dvi_StackSize;
    LONG  dvi_Priority;
    LONG  dvi_Startup;
    BPTR  dvi_SegList;
    BPTR  dvi_GlobVec;
    BSTR  dvi_Name;
};

/* combined structure for devices, assigned directories, volumes */

struct DosList {
    BPTR                dol_Next;        /* bptr to next device on list */
    LONG                dol_Type;        /* see DLT below */
    struct MsgPort     *dol_Task;        /* ptr to handler task */
    BPTR                dol_Lock;
    union {
        struct {
        BSTR    dol_Handler;    /* file name to load if seglist is null */
        LONG    dol_StackSize;  /* stacksize to use when starting process */
        LONG    dol_Priority;   /* task priority when starting process */
        ULONG   dol_Startup;    /* startup msg: FileSysStartupMsg for disks */
        BPTR    dol_SegList;    /* already loaded code for new task */
        BPTR    dol_GlobVec;    /* BCPL global vector to use when starting
                                 * a process. -1 indicates a C/Assembler
                                 * program. */
        } dol_handler;

        struct {
        struct DateStamp        dol_VolumeDate;  /* creation date */
        BPTR                    dol_LockList;    /* outstanding locks */
        LONG                    dol_DiskType;    /* 'DOS', etc */
        } dol_volume;

        struct {
        UBYTE   *dol_AssignName;     /* name for non-or-late-binding assign */
        struct AssignList *dol_List; /* for multi-directory assigns (regular) */
        } dol_assign;

    } dol_misc;

    BSTR                dol_Name;        /* bptr to bcpl name */
    };

/* structure used for multi-directory assigns. AllocVec()ed. */

```

```
struct AssignList {
        struct AssignList *al_Next;
        BPTR              al_Lock;
};

/* definitions for dl_Type */
#define DLT_DEVICE      0
#define DLT_DIRECTORY   1       /* assign */
#define DLT_VOLUME      2
#define DLT_LATE        3       /* late-binding assign */
#define DLT_NONBINDING  4       /* non-binding assign */
#define DLT_PRIVATE     -1      /* for internal use only */

/* structure return by GetDeviceProc() */
struct DevProc {
        struct MsgPort *dvp_Port;
        BPTR            dvp_Lock;
        ULONG           dvp_Flags;
        struct DosList *dvp_DevNode;    /* DON'T TOUCH OR USE! */
};

/* definitions for dvp_Flags */
#define DVPB_UNLOCK     0
#define DVPF_UNLOCK     (1L << DVPB_UNLOCK)
#define DVPB_ASSIGN     1
#define DVPF_ASSIGN     (1L << DVPB_ASSIGN)

/* Flags to be passed to LockDosList(), etc */
#define LDB_DEVICES     2
#define LDF_DEVICES     (1L << LDB_DEVICES)
#define LDB_VOLUMES     3
#define LDF_VOLUMES     (1L << LDB_VOLUMES)
#define LDB_ASSIGNS     4
#define LDF_ASSIGNS     (1L << LDB_ASSIGNS)
#define LDB_ENTRY       5
#define LDF_ENTRY       (1L << LDB_ENTRY)
#define LDB_DELETE      6
#define LDF_DELETE      (1L << LDB_DELETE)

/* you MUST specify one of LDF_READ or LDF_WRITE */
#define LDB_READ        0
#define LDF_READ        (1L << LDB_READ)
#define LDB_WRITE       1
#define LDF_WRITE       (1L << LDB_WRITE)

/* actually all but LDF_ENTRY (which is used for internal locking) */
#define LDF_ALL         (LDF_DEVICES|LDF_VOLUMES|LDF_ASSIGNS)

/* a lock structure, as returned by Lock() or DupLock() */
struct FileLock {
    BPTR                fl_Link;        /* bcpl pointer to next lock */
    LONG                fl_Key;         /* disk block number */
    LONG                fl_Access;      /* exclusive or shared */
    struct MsgPort *    fl_Task;        /* handler task's port */
    BPTR                fl_Volume;      /* bptr to DLT_VOLUME DosList entry */
};

/* error report types for ErrorReport() */
#define REPORT_STREAM           0       /* a stream */
#define REPORT_TASK             1       /* a process - unused */
#define REPORT_LOCK             2       /* a lock */
#define REPORT_VOLUME           3       /* a volume node */
#define REPORT_INSERT           4       /* please insert volume */

/* Special error codes for ErrorReport() */
#define ABORT_DISK_ERROR        296     /* Read/write error */
```

```
#define ABORT_BUSY              288     /* You MUST replace... */

/* types for initial packets to shells from run/newcli/execute/system. */
/* For shell-writers only */
#define RUN_EXECUTE             -1
#define RUN_SYSTEM              -2
#define RUN_SYSTEM_ASYNCH       -3

/* Types for fib_DirEntryType.  NOTE that both USERDIR and ROOT are      */
/* directories, and that directory/file checks should use <0 and >=0.    */
/* This is not necessarily exhaustive!  Some handlers may use other      */
/* values as needed, though <0 and >=0 should remain as supported as     */
/* possible.                                                             */
#define ST_ROOT         1
#define ST_USERDIR      2
#define ST_SOFTLINK     3       /* looks like dir, but may point to a file! */
#define ST_LINKDIR      4       /* hard link to dir */
#define ST_FILE         -3      /* must be negative for FIB! */
#define ST_LINKFILE     -4      /* hard link to file */

#endif  /* DOS_DOSEXTENS_H */
```

dos/doshunks.h

Page 1

```
#ifndef DOS_DOSHUNKS_H
#define DOS_DOSHUNKS_H
/*
**      $Filename: dos/doshunks.h $
**      $Release: 2.04 $
**      $Revision: 36.5 $
**      $Date: 90/07/12 $
**
**      Hunk definitions for object and load modules.
**
**      (C) Copyright 1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

/* hunk types */
#define HUNK_UNIT       999
#define HUNK_NAME       1000
#define HUNK_CODE       1001
#define HUNK_DATA       1002
#define HUNK_BSS        1003
#define HUNK_RELOC32    1004
#define HUNK_RELOC16    1005
#define HUNK_RELOC8     1006
#define HUNK_EXT        1007
#define HUNK_SYMBOL     1008
#define HUNK_DEBUG      1009
#define HUNK_END        1010
#define HUNK_HEADER     1011

#define HUNK_OVERLAY    1013
#define HUNK_BREAK      1014

#define HUNK_DREL32     1015
#define HUNK_DREL16     1016
#define HUNK_DREL8      1017

#define HUNK_LIB        1018
#define HUNK_INDEX      1019

/* hunk_ext sub-types */
#define EXT_SYMB        0       /* symbol table */
#define EXT_DEF         1       /* relocatable definition */
#define EXT_ABS         2       /* Absolute definition */
#define EXT_RES         3       /* no longer supported */
#define EXT_REF32       129     /* 32 bit reference to symbol */
#define EXT_COMMON      130     /* 32 bit reference to COMMON block */
#define EXT_REF16       131     /* 16 bit reference to symbol */
#define EXT_REF8        132     /*  8 bit reference to symbol */
#define EXT_DEXT32      133     /* 32 bit data releative reference */
#define EXT_DEXT16      134     /* 16 bit data releative reference */
#define EXT_DEXT8       135     /*  8 bit data releative reference */

#endif  /* DOS_DOSHUNKS_H */
```

dos/dostags.h

Page 1

```
#ifndef DOS_DOSTAGS_H
#define DOS_DOSTAGS_H
/*
**
**      $Filename: dos/dostags.h $
**      $Release: 2.04 $
**      $Revision: 36.10 $
**      $Date: 90/11/02 $
**
**      Tag definitions for all Dos routines using tags
**
**      (C) Copyright 1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**
*/

#ifndef UTILITY_TAGITEM_H
#include "utility/tagitem.h"
#endif

/*****************************************************************************/
/* definitions for the System() call */

#define SYS_Dummy       (TAG_USER + 32)
#define SYS_Input       (SYS_Dummy + 1)
                                /* specifies the input filehandle  */
#define SYS_Output      (SYS_Dummy + 2)
                                /* specifies the output filehandle */
#define SYS_Asynch      (SYS_Dummy + 3)
                                /* run asynch, close input/output on exit(!) */
#define SYS_UserShell   (SYS_Dummy + 4)
                                /* send to user shell instead of boot shell */
#define SYS_CustomShell (SYS_Dummy + 5)
                                /* send to a specific shell (data is name) */
/*      SYS_Error, */


/*****************************************************************************/
/* definitions for the CreateNewProc() call */
/* you MUST specify one of NP_Seglist or NP_Entry.  All else is optional. */

#define NP_Dummy (TAG_USER + 1000)
#define NP_Seglist      (NP_Dummy + 1)
                                /* seglist of code to run for the process  */
#define NP_FreeSeglist  (NP_Dummy + 2)
                                /* free seglist on exit - only valid for   */
                                /* for NP_Seglist.  Default is TRUE.       */
#define NP_Entry        (NP_Dummy + 3)
                                /* entry point to run - mutually exclusive */
                                /* with NP_Seglist! */
#define NP_Input        (NP_Dummy + 4)
                                /* filehandle - default is Open("NIL:"...) */
#define NP_Output       (NP_Dummy + 5)
                                /* filehandle - default is Open("NIL:"...) */
#define NP_CloseInput   (NP_Dummy + 6)
                                /* close input filehandle on exit          */
                                /* default TRUE                            */
#define NP_CloseOutput  (NP_Dummy + 7)
                                /* close output filehandle on exit         */
                                /* default TRUE                            */
#define NP_Error        (NP_Dummy + 8)
                                /* filehandle - default is Open("NIL:"...) */
#define NP_CloseError   (NP_Dummy + 9)
                                /* close error filehandle on exit          */
                                /* default TRUE                            */
#define NP_CurrentDir   (NP_Dummy + 10)
```

```
                                /* lock - default is parent's current dir */
#define NP_StackSize    (NP_Dummy + 11)
                                /* stacksize for process - default 4000    */
#define NP_Name         (NP_Dummy + 12)
                                /* name for process - default "New Process"*/
#define NP_Priority     (NP_Dummy + 13)
                                /* priority - default same as parent       */
#define NP_ConsoleTask  (NP_Dummy + 14)
                                /* consoletask - default same as parent    */
#define NP_WindowPtr    (NP_Dummy + 15)
                                /* window ptr - default is same as parent  */
#define NP_HomeDir      (NP_Dummy + 16)
                                /* home directory - default curr home dir  */
#define NP_CopyVars     (NP_Dummy + 17)
                                /* boolean to copy local vars-default TRUE */
#define NP_Cli          (NP_Dummy + 18)
                                /* create cli structure - default FALSE    */
#define NP_Path         (NP_Dummy + 19)
                                /* path - default is copy of parents path  */
                                /* only valid if a cli process!    */
#define NP_CommandName  (NP_Dummy + 20)
                                /* commandname - valid only for CLI        */
#define NP_Arguments    (NP_Dummy + 21)
                                /* cstring of arguments - passed with str  */
                                /* in a0, length in d0.  (copied and freed */
                                /* on exit.  Default is empty string.      */
                                /* NOTE: not operational until 2.04 - see  */
                                /* BIX/TechNotes for more info/workarounds */
                                /* NOTE: in 2.0, it DIDN'T pass "" - the   */
                                /* registers were random.                  */
/* FIX! should this be only for cli's? */
#define NP_NotifyOnDeath (NP_Dummy + 22)
                                /* notify parent on death - default FALSE  */
                                /* Not functional yet. */
#define NP_Synchronous  (NP_Dummy + 23)
                                /* don't return until process finishes -   */
                                /* default FALSE.                          */
                                /* Not functional yet. */
#define NP_ExitCode     (NP_Dummy + 24)
                                /* code to be called on process exit       */
#define NP_ExitData     (NP_Dummy + 25)
                                /* optional argument for NP_EndCode rtn -  */
                                /* default NULL                            */


/**************************************************************************/
/* tags for AllocDosObject */

#define ADO_Dummy       (TAG_USER + 2000)
#define ADO_FH_Mode     (ADO_Dummy + 1)
                                /* for type DOS_FILEHANDLE only            */
                                /* sets up FH for mode specified.
                                   This can make a big difference for buffered
                                   files.                                  */
        /* The following are for DOS_CLI */
        /* If you do not specify these, dos will use it's preferred values */
        /* which may change from release to release.  The BPTRs to these   */
        /* will be set up correctly for you.  Everything will be zero,     */
        /* except cli_FailLevel (10) and cli_Background (DOSTRUE).         */
        /* NOTE: you may also use these 4 tags with CreateNewProc.         */

#define ADO_DirLen      (ADO_Dummy + 2)
                                /* size in bytes for current dir buffer    */
#define ADO_CommNameLen (ADO_Dummy + 3)
                                /* size in bytes for command name buffer   */
#define ADO_CommFileLen (ADO_Dummy + 4)
```

```
                                /* size in bytes for command file buffer   */
#define ADO_PromptLen   (ADO_Dummy + 5)
                                /* size in bytes for the prompt buffer     */

/**************************************************************************/
/* tags for NewLoadSeg */
/* no tags are defined yet for NewLoadSeg */

#endif /* DOS_DOSTAGS_H */
```

dos/exall.h

Page 1

```
#ifndef DOS_EXALL_H
#define DOS_EXALL_H
/*
**      $Filename: dos/exall.h $
**      $Release: 2.04 $
**      $Revision: 36.5 $
**      $Date: 90/07/12 $
**
**      include file for ExAll() data structures
**
**      (C) Copyright 1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_TYPES_H
#include "exec/types.h"
#endif

#ifndef UTILITY_HOOKS_H
#include "utility/hooks.h"
#endif

/* values that can be passed for what data you want from ExAll() */
/* each higher value includes those below it (numerically)       */
/* you MUST chose one of these values */
#define ED_NAME         1
#define ED_TYPE         2
#define ED_SIZE         3
#define ED_PROTECTION   4
#define ED_DATE         5
#define ED_COMMENT      6

/*   Structure in which exall results are returned in.  Note that only the
 *   fields asked for will exist!
 */

struct ExAllData {
        struct ExAllData *ed_Next;
        UBYTE  *ed_Name;
        LONG    ed_Type;
        ULONG   ed_Size;
        ULONG   ed_Prot;
        ULONG   ed_Days;
        ULONG   ed_Mins;
        ULONG   ed_Ticks;
        UBYTE  *ed_Comment;     /* strings will be after last used field */
};

/*   Control structure passed to ExAll.  Unused fields MUST be initialized to
 *   0, expecially eac_LastKey.
 *
 *   eac_MatchFunc is a hook (see utility.library documentation for usage)
 *   It should return true if the entry is to returned, false if it is to be
 *   ignored.
 *
 *   This structure MUST be allocated by AllocDosObject()!
 */

struct ExAllControl {
        ULONG   eac_Entries;     /* number of entries returned in buffer      */
        ULONG   eac_LastKey;     /* Don't touch inbetween linked ExAll calls! */
        UBYTE  *eac_MatchString; /* wildcard string for pattern match or NULL */
        struct Hook *eac_MatchFunc; /* optional private wildcard function     */
};

#endif /* DOS_EXALL_H */
```

dos/filehandler.h

Page 1

```
#ifndef DOS_FILEHANDLER_H
#define DOS_FILEHANDLER_H
/*
**      $Filename: dos/filehandler.h $
**      $Release: 2.04 $
**      $Revision: 36.5 $
**      $Date: 90/07/12 $
**
**      device and file handler specific code for AmigaDOS
**
**      (C) Copyright 1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef   EXEC_PORTS_H
#include "exec/ports.h"
#endif

#ifndef   DOS_DOS_H
#include "dos/dos.h"
#endif


/* The disk "environment" is a longword array that describes the
 * disk geometry.  It is variable sized, with the length at the beginning.
 * Here are the constants for a standard geometry.
 */

struct DosEnvec {
    ULONG de_TableSize;      /* Size of Environment vector */
    ULONG de_SizeBlock;      /* in longwords: standard value is 128 */
    ULONG de_SecOrg;         /* not used; must be 0 */
    ULONG de_Surfaces;       /* # of heads (surfaces). drive specific */
    ULONG de_SectorPerBlock; /* not used; must be 1 */
    ULONG de_BlocksPerTrack; /* blocks per track. drive specific */
    ULONG de_Reserved;       /* DOS reserved blocks at start of partition. */
    ULONG de_PreAlloc;       /* DOS reserved blocks at end of partition */
    ULONG de_Interleave;     /* usually 0 */
    ULONG de_LowCyl;         /* starting cylinder. typically 0 */
    ULONG de_HighCyl;        /* max cylinder. drive specific */
    ULONG de_NumBuffers;     /* Initial # DOS of buffers.  */
    ULONG de_BufMemType;     /* type of mem to allocate for buffers */
    ULONG de_MaxTransfer;    /* Max number of bytes to transfer at a time */
    ULONG de_Mask;           /* Address Mask to block out certain memory */
    LONG  de_BootPri;        /* Boot priority for autoboot */
    ULONG de_DosType;        /* ASCII (HEX) string showing filesystem type;
                              * 0X444F5300 is old filesystem,
                              * 0X444F5301 is fast file system */
    ULONG de_Baud;           /* Baud rate for serial handler */
    ULONG de_Control;        /* Control word for handler/filesystem */
    ULONG de_BootBlocks;     /* Number of blocks containing boot code */

};

/* these are the offsets into the array */

#define DE_TABLESIZE    0       /* standard value is 11 */
#define DE_SIZEBLOCK    1       /* in longwords: standard value is 128 */
#define DE_SECORG       2       /* not used; must be 0 */
#define DE_NUMHEADS     3       /* # of heads (surfaces). drive specific */
#define DE_SECSPERBLK   4       /* not used; must be 1 */
#define DE_BLKSPERTRACK 5       /* blocks per track. drive specific */
#define DE_RESERVEDBLKS 6       /* unavailable blocks at start.  usually 2 */
#define DE_PREFAC       7       /* not used; must be 0 */
#define DE_INTERLEAVE   8       /* usually 0 */
#define DE_LOWCYL       9       /* starting cylinder. typically 0 */
```

dos/filehandler.h

```
#define DE_UPPERCYL     10      /* max cylinder.  drive specific */
#define DE_NUMBUFFERS   11      /* starting # of buffers.  typically 5 */
#define DE_MEMBUFTYPE   12      /* type of mem to allocate for buffers. */
#define DE_BUFMEMTYPE   12      /* same as above, better name
                                 * 1 is public, 3 is chip, 5 is fast */
#define DE_MAXTRANSFER  13      /* Max number bytes to transfer at a time */
#define DE_MASK         14      /* Address Mask to block out certain memory */
#define DE_BOOTPRI      15      /* Boot priority for autoboot */
#define DE_DOSTYPE      16      /* ASCII (HEX) string showing filesystem type;
                                 * 0X444F5300 is old filesystem,
                                 * 0X444F5301 is fast file system */
#define DE_BAUD         17      /* Baud rate for serial handler */
#define DE_CONTROL      18      /* Control word for handler/filesystem */
#define DE_BOOTBLOCKS   19      /* Number of blocks containing boot code */

/* The file system startup message is linked into a device node's startup
** field.  It contains a pointer to the above environment, plus the
** information needed to do an exec OpenDevice().
*/
struct FileSysStartupMsg {
    ULONG       fssm_Unit;      /* exec unit number for this device */
    BSTR        fssm_Device;    /* null terminated bstring to the device name */
    BPTR        fssm_Environ;   /* ptr to environment table (see above) */
    ULONG       fssm_Flags;     /* flags for OpenDevice() */
};


/* The include file "libraries/dosextens.h" has a DeviceList structure.
 * The "device list" can have one of three different things linked onto
 * it.  Dosextens defines the structure for a volume.  DLT_DIRECTORY
 * is for an assigned directory.  The following structure is for
 * a dos "device" (DLT_DEVICE).
*/

struct DeviceNode {
    BPTR        dn_Next;        /* singly linked list */
    ULONG       dn_Type;        /* always 0 for dos "devices" */
    struct MsgPort *dn_Task;    /* standard dos "task" field.  If this is
                                 * null when the node is accesses, a task
                                 * will be started up */
    BPTR        dn_Lock;        /* not used for devices -- leave null */
    BSTR        dn_Handler;     /* filename to loadseg (if seglist is null) */
    ULONG       dn_StackSize;   /* stacksize to use when starting task */
    LONG        dn_Priority;    /* task priority when starting task */
    BPTR        dn_Startup;     /* startup msg: FileSysStartupMsg for disks */
    BPTR        dn_SegList;     /* code to run to start new task (if necessary).
                                 * if null then dn_Handler will be loaded. */
    BPTR        dn_GlobalVec;   /* BCPL global vector to use when starting
                                 * a task.  -1 means that dn_SegList is not
                                 * for a bcpl program, so the dos won't
                                 * try and construct one.  0 tell the
                                 * dos that you obey BCPL linkage rules,
                                 * and that it should construct a global
                                 * vector for you.
                                 */
    BSTR        dn_Name;        /* the node name, e.g. '\3','D','F','3' */
};

#endif  /* DOS_FILEHANDLER_H */
```

dos/notify.h

```
#ifndef DOS_NOTIFY_H
#define DOS_NOTIFY_H
/*
**
**      $Filename: dos/notify.h $
**      $Release: 2.04 $
**      $Revision: 36.8 $
**      $Date: 90/08/29 $
**
**      dos notification definitions
**
**      (C) Copyright 1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**
*/

#ifndef EXEC_TYPES_H
#include "exec/types.h"
#endif

#ifndef EXEC_PORTS_H
#include "exec/ports.h"
#endif

#ifndef EXEC_TASKS_H
#include "exec/tasks.h"
#endif


/* use of Class and code is discouraged for the time being - we might want to
   change things */
/* --- NotifyMessage Class ------------------------------------------------ */
#define NOTIFY_CLASS    0x40000000

/* --- NotifyMessage Codes ------------------------------------------------ */
#define NOTIFY_CODE     0x1234


/* Sent to the application if SEND_MESSAGE is specified.                  */

struct NotifyMessage {
    struct Message nm_ExecMessage;
    ULONG  nm_Class;
    UWORD  nm_Code;
    struct NotifyRequest *nm_NReq;      /* don't modify the request! */
    ULONG  nm_DoNotTouch;               /* like it says!  For use by handlers */
    ULONG  nm_DoNotTouch2;              /* ditto */
};

/* Do not modify or reuse the notifyrequest while active.                 */
/* note: the first LONG of nr_Data has the length transfered              */

struct NotifyRequest {
        UBYTE *nr_Name;
        UBYTE *nr_FullName;             /* set by dos - don't touch */
        ULONG nr_UserData;              /* for applications use */
        ULONG nr_Flags;

        union {

            struct {
                struct MsgPort *nr_Port;        /* for SEND_MESSAGE */
            } nr_Msg;

            struct {
                struct Task *nr_Task;           /* for SEND_SIGNAL */
```

dos/notify.h

```
                UBYTE nr_SignalNum;             /* for SEND_SIGNAL */
                UBYTE nr_pad[3];
            } nr_Signal;
        } nr_stuff;

        ULONG nr_Reserved[4];           /* leave 0 for now */

        /* internal use by handlers */
        ULONG nr_MsgCount;              /* # of outstanding msgs */
        struct MsgPort *nr_Handler;     /* handler sent to (for EndNotify) */
};

/* --- NotifyRequest Flags ------------------------------------------- */
#define NRF_SEND_MESSAGE        1
#define NRF_SEND_SIGNAL         2
#define NRF_WAIT_REPLY          8
#define NRF_NOTIFY_INITIAL      16

/* do NOT set or remove NRF_MAGIC!  Only for use by handlers! */
#define NRF_MAGIC       0x80000000

/* bit numbers */
#define NRB_SEND_MESSAGE        0
#define NRB_SEND_SIGNAL         1
#define NRB_WAIT_REPLY          3
#define NRB_NOTIFY_INITIAL      4

#define NRB_MAGIC               31

/* Flags reserved for private use by the handler: */
#define NR_HANDLER_FLAGS        0xffff0000

#endif /* DOS_NOTIFY_H */
```

dos/rdargs.h

```
#ifndef DOS_RDARGS_H
#define DOS_RDARGS_H
/*
**
**      $Filename: dos/rdargs.h $
**      $Release: 2.04 $
**      $Revision: 36.6 $
**      $Date: 90/07/12 $
**
**      ReadArgs() structure definitions
**
**      (C) Copyright 1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**
*/

#ifndef EXEC_TYPES_H
#include "exec/types.h"
#endif

#ifndef EXEC_NODES_H
#include "exec/nodes.h"
#endif

/**********************************************************************
 *
 * The CSource data structure defines the input source for "ReadItem()"
 * as well as the ReadArgs call.  It is a publicly defined structure
 * which may be used by applications which use code that follows the
 * conventions defined for access.
 *
 * When passed to the dos.library functions, the value passed as
 * struct *CSource is defined as follows:
 *      if ( CSource == 0)      Use buffered IO "ReadChar()" as data source
 *      else                    Use CSource for input character stream
 *
 * The following two pseudo-code routines define how the CSource structure
 * is used:
 *
 * long CS_ReadChar( struct CSource *CSource )
 * {
 *      if ( CSource == 0 )     return ReadChar();
 *      if ( CSource->CurChr >= CSource->Length )       return ENDSTREAMCHAR;
 *      return CSource->Buffer[ CSource->CurChr++ ];
 * }
 *
 * BOOL CS_UnReadChar( struct CSource *CSource )
 * {
 *      if ( CSource == 0 )     return UnReadChar();
 *      if ( CSource->CurChr <= 0 )     return FALSE;
 *      CSource->CurChr--;
 *      return TRUE;
 * }
 *
 * To initialize a struct CSource, you set CSource->CS_Buffer to
 * a string which is used as the data source, and set CS_Length to
 * the number of characters in the string.  Normally CS_CurChr should
 * be initialized to ZERO, or left as it was from prior use as
 * a CSource.
 *
 **********************************************************************/

struct CSource {
        UBYTE   *CS_Buffer;
        LONG    CS_Length;
        LONG    CS_CurChr;
```

dos/rdargs.h

```
};

/***********************************************************************
 *
 * The RDArgs data structure is the input parameter passed to the DOS
 * ReadArgs() function call.
 *
 * The RDA_Source structure is a CSource as defined above;
 * if RDA_Source.CS_Buffer is non-null, RDA_Source is used as the input
 * character stream to parse, else the input comes from the buffered STDIN
 * calls ReadChar/UnReadChar.
 *
 * RDA_DAList is a private address which is used internally to track
 * allocations which are freed by FreeArgs().  This MUST be initialized
 * to NULL prior to the first call to ReadArgs().
 *
 * The RDA_Buffer and RDA_BufSiz fields allow the application to supply
 * a fixed-size buffer in which to store the parsed data.  This allows
 * the application to pre-allocate a buffer rather than requiring buffer
 * space to be allocated.  If either RDA_Buffer or RDA_BufSiz is NULL,
 * the application has not supplied a buffer.
 *
 * RDA_ExtHelp is a text string which will be displayed instead of the
 * template string, if the user is prompted for input.
 *
 * RDA_Flags bits control how ReadArgs() works.  The flag bits are
 * defined below.  Defaults are initialized to ZERO.
 *
 ***********************************************************************/

struct RDArgs {
        struct  CSource RDA_Source;     /* Select input source */
        LONG    RDA_DAList;             /* PRIVATE. */
        UBYTE   *RDA_Buffer;            /* Optional string parsing space. */
        LONG    RDA_BufSiz;             /* Size of RDA_Buffer (0..n) */
        UBYTE   *RDA_ExtHelp;           /* Optional extended help */
        LONG    RDA_Flags;              /* Flags for any required control */
};

#define RDAB_STDIN      0       /* Use "STDIN" rather than "COMMAND LINE" */
#define RDAF_STDIN      1
#define RDAB_NOALLOC    1       /* If set, do not allocate extra string space.*/
#define RDAF_NOALLOC    2
#define RDAB_NOPROMPT   2       /* Disable reprompting for string input. */
#define RDAF_NOPROMPT   4

/***********************************************************************
 * Maximum number of template keywords which can be in a template passed
 * to ReadArgs(). IMPLEMENTOR NOTE - must be a multiple of 4.
 ***********************************************************************/
#define MAX_TEMPLATE_ITEMS      100

/***********************************************************************
 * Maximum number of MULTIARG items returned by ReadArgs(), before
 * an ERROR_LINE_TOO_LONG.  These two limitations are due to stack
 * usage.  Applications should allow "a lot" of stack to use ReadArgs().
 ***********************************************************************/
#define MAX_MULTIARGS           128

#endif /* DOS_RDARGS_H */
```

dos/record.h

```
#ifndef DOS_RECORD_H
#define DOS_RECORD_H
/*
**
**      $Filename: dos/record.h $
**      $Release: 2.04 $
**      $Revision: 36.5 $
**      $Date: 90/07/12 $
**
**      include file for record locking
**
**      (C) Copyright 1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**
*/

#ifndef DOS_DOS_H
#include "dos/dos.h"
#endif

/* Modes for LockRecord/LockRecords() */
#define REC_EXCLUSIVE           0
#define REC_EXCLUSIVE_IMMED     1
#define REC_SHARED              2
#define REC_SHARED_IMMED        3

/* struct to be passed to LockRecords()/UnLockRecords() */

struct RecordLock {
        BPTR    rec_FH;         /* filehandle */
        ULONG   rec_Offset;     /* offset in file */
        ULONG   rec_Length;     /* length of file to be locked */
        ULONG   rec_Mode;       /* Type of lock */
};

#endif /* DOS_RECORD_H */
```

dos/stdio.h

```c
#ifndef DOS_STDIO_H
#define DOS_STDIO_H
/*
**
**      $Filename: dos/stdio.h $
**      $Release: 2.04 $
**      $Revision: 36.5 $
**      $Date: 90/11/05 $
**
**      ANSI-like stdio defines for dos buffered I/O
**
**      (C) Copyright 1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**
*/

#define ReadChar()              FGetC(Input())
#define WriteChar(c)            FPutC(Output(),(c))
#define UnReadChar(c)           UnGetC(Input(),(c))
/* next one is inefficient */
#define ReadChars(buf,num)      FRead(Input(),(buf),1,(num))
#define ReadLn(buf,len)         FGets(Input(),(buf),(len))
#define WriteStr(s)             FPuts(Output(),(s))
#define VWritef(format,argv)    VFWritef(Output(),(format),(argv))

/* types for SetVBuf */
#define BUF_LINE        0       /* flush on \n, etc */
#define BUF_FULL        1       /* never flush except when needed */
#define BUF_NONE        2       /* no buffering */

#endif  /* DOS_STDIO_H */
```

dos/var.h

```c
#ifndef DOS_VAR_H
#define DOS_VAR_H
/*
**
**      $Filename: dos/var.h $
**      $Release: 2.04 $
**      $Revision: 36.9 $
**      $Date: 90/07/12 $
**
**      include file for dos local and environment variables
**
**      (C) Copyright 1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**
*/


#ifndef EXEC_NODES_H
#include "exec/nodes.h"
#endif

/* the structure in the pr_LocalVars list */
/* Do NOT allocate yourself, use SetVar()!!! This structure may grow in */
/* future releases!  The list should be left in alphabetical order, and */
/* may have multiple entries with the same name but different types.    */

struct LocalVar {
        struct Node lv_Node;
        UWORD   lv_Flags;
        UBYTE   *lv_Value;
        ULONG   lv_Len;
};

/*
 * The lv_Flags bits are available to the application.  The unused
 * lv_Node.ln_Pri bits are reserved for system use.
 */

/* bit definitions for lv_Node.ln_Type: */
#define LV_VAR                  0       /* an variable */
#define LV_ALIAS                1       /* an alias */
/* to be or'ed into type: */
#define LVB_IGNORE              7       /* ignore this entry on GetVar, etc */
#define LVF_IGNORE              0x80

/* definitions of flags passed to GetVar()/SetVar()/DeleteVar() */
/* bit defs to be OR'ed with the type: */
/* item will be treated as a single line of text unless BINARY_VAR is used */
#define GVB_GLOBAL_ONLY         8
#define GVF_GLOBAL_ONLY         0x100
#define GVB_LOCAL_ONLY          9
#define GVF_LOCAL_ONLY          0x200
#define GVB_BINARY_VAR          10              /* treat variable as binary */
#define GVF_BINARY_VAR          0x400

#endif /* DOS_VAR_H */
```

```
#ifndef EXEC_ALERTS_H
#define EXEC_ALERTS_H
/*
**      $Filename: exec/alerts.h $
**      $Release: 2.04 $
**      $Revision: 36.18 $
**      $Date: 91/01/12 $
**
**      Alert numbers, as displayed by system crashes.
**
**      (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

/*********************************************************************
*
*  Format of the alert error number:
*
*    +-+-------------+----------------+--------------------------------+
*    |D|  SubSysId   |  General Error |    SubSystem Specific Error    |
*    +-+-------------+----------------+--------------------------------+
*     1    7 bits          8 bits                  16 bits
*
*                    D:  DeadEnd alert
*             SubSysId:  indicates ROM subsystem number.
*        General Error:  roughly indicates what the error was
*       Specific Error:  indicates more detail
**********************************************************************/

/*********************************************************************
*
*  General Alerts
*
*  For example: timer.device cannot open math.library would be 0x05038015
*
*       Alert(AN_TimerDev|AG_OpenLib|AO_MathLib);
*
**********************************************************************/

/*------ alert types */
#define AT_DeadEnd      0x80000000
#define AT_Recovery     0x00000000


/*------ general purpose alert codes */
#define AG_NoMemory     0x00010000
#define AG_MakeLib      0x00020000
#define AG_OpenLib      0x00030000
#define AG_OpenDev      0x00040000
#define AG_OpenRes      0x00050000
#define AG_IOError      0x00060000
#define AG_NoSignal     0x00070000
#define AG_BadParm      0x00080000
#define AG_CloseLib     0x00090000      /* usually too many closes */
#define AG_CloseDev     0x000A0000      /* or a mismatched close */
#define AG_ProcCreate   0x000B0000      /* Process creation failed */


/*------ alert objects: */
#define AO_ExecLib      0x00008001
#define AO_GraphicsLib  0x00008002
#define AO_LayersLib    0x00008003
#define AO_Intuition    0x00008004
#define AO_MathLib      0x00008005
#define AO_DOSLib       0x00008007
#define AO_RAMLib       0x00008008
```

```
#define AO_IconLib      0x00008009
#define AO_ExpansionLib 0x0000800A
#define AO_DiskfontLib  0x0000800B
#define AO_UtilityLib   0x0000800C

#define AO_AudioDev     0x00008010
#define AO_ConsoleDev   0x00008011
#define AO_GamePortDev  0x00008012
#define AO_KeyboardDev  0x00008013
#define AO_TrackDiskDev 0x00008014
#define AO_TimerDev     0x00008015

#define AO_CIARsrc      0x00008020
#define AO_DiskRsrc     0x00008021
#define AO_MiscRsrc     0x00008022

#define AO_BootStrap    0x00008030
#define AO_Workbench    0x00008031
#define AO_DiskCopy     0x00008032
#define AO_GadTools     0x00008033
#define AO_Unknown      0x00008035



/*********************************************************************
*
*   Specific Alerts:
*
**********************************************************************/

/*------ exec.library */
#define AN_ExecLib      0x01000000
#define AN_ExcptVect    0x01000001 /* 68000 exception vector checksum (obs.) */
#define AN_BaseChkSum   0x01000002 /* Execbase checksum (obs.) */
#define AN_LibChkSum    0x01000003 /* Library checksum failure */

#define AN_MemCorrupt   0x81000005 /* Corrupt memory list detected in FreeMem */
#define AN_IntrMem      0x81000006 /* No memory for interrupt servers */
#define AN_InitAPtr     0x01000007 /* InitStruct() of an APTR source (obs.) */
#define AN_SemCorrupt   0x01000008 /* A semaphore is in an illegal state
                                      at ReleaseSempahore() */
#define AN_FreeTwice    0x01000009 /* Freeing memory already freed */
#define AN_BogusExcpt   0x8100000A /* illegal 68k exception taken (obs.) */
#define AN_IOUsedTwice  0x0100000B /* Attempt to reuse active IORequest */
#define AN_MemoryInsane 0x0100000C /* Sanity check on memory list failed
                                      during AvailMem(MEMF_LARGEST) */
#define AN_IOAfterClose 0x0100000D /* IO attempted on closed IORequest */
#define AN_StackProbe   0x0100000E /* Stack appears to extend out of range */
#define AN_BadFreeAddr  0x0100000F /* Memory header not located. [ Usually an
                                      invalid address passed to FreeMem() ] */

/*------ graphics.library */
#define AN_GraphicsLib  0x02000000
#define AN_GfxNoMem     0x82010000      /* graphics out of memory */
#define AN_GfxNoMemMspc 0x82010001      /* MonitorSpec alloc, no memory */
#define AN_LongFrame    0x82010006      /* long frame, no memory */
#define AN_ShortFrame   0x82010007      /* short frame, no memory */
#define AN_TextTmpRas   0x02010009      /* text, no memory for TmpRas */
#define AN_BltBitMap    0x8201000A      /* BltBitMap, no memory */
#define AN_RegionMemory 0x8201000B      /* regions, memory not available */
#define AN_MakeVPort    0x82010030      /* MakeVPort, no memory */
#define AN_GfxNewError  0x0200000C
#define AN_GfxFreeError 0x0200000D

#define AN_GfxNoLCM     0x82011234      /* emergency memory not available */

```

exec/alerts.h

Page 3

```
#define AN_ObsoleteFont 0x02000401      /* unsupported font description used */

/*------ layers.library */
#define AN_LayersLib    0x03000000
#define AN_LayersNoMem  0x83010000      /* layers out of memory */

/*------ intuition.library */
#define AN_Intuition    0x04000000
#define AN_GadgetType   0x84000001      /* unknown gadget type */
#define AN_BadGadget    0x04000001      /* Recovery form of AN_GadgetType */
#define AN_CreatePort   0x84010002      /* create port, no memory */
#define AN_ItemAlloc    0x04010003      /* item plane alloc, no memory */
#define AN_SubAlloc     0x04010004      /* sub alloc, no memory */
#define AN_PlaneAlloc   0x84010005      /* plane alloc, no memory */
#define AN_ItemBoxTop   0x84000006      /* item box top < RelZero */
#define AN_OpenScreen   0x84010007      /* open screen, no memory */
#define AN_OpenScrnRast 0x84010008      /* open screen, raster alloc, no memory
   */
#define AN_SysScrnType  0x84000009      /* open sys screen, unknown type */
#define AN_AddSWGadget  0x8401000A      /* add SW gadgets, no memory */
#define AN_OpenWindow   0x8401000B      /* open window, no memory */
#define AN_BadState     0x8400000C      /* Bad State Return entering Intuition *
  /
#define AN_BadMessage   0x8400000D      /* Bad Message received by IDCMP */
#define AN_WeirdEcho    0x8400000E      /* Weird echo causing incomprehension */
#define AN_NoConsole    0x8400000F      /* couldn't open the Console Device */

/*------ math.library */
#define AN_MathLib      0x05000000

/*------ dos.library */
#define AN_DOSLib       0x07000000
#define AN_StartMem     0x07010001 /* no memory at startup */
#define AN_EndTask      0x07000002 /* EndTask didn't */
#define AN_QPktFail     0x07000003 /* Qpkt failure */
#define AN_AsyncPkt     0x07000004 /* Unexpected packet received */
#define AN_FreeVec      0x07000005 /* Freevec failed */
#define AN_DiskBlkSeq   0x07000006 /* Disk block sequence error */
#define AN_BitMap       0x07000007 /* Bitmap corrupt */
#define AN_KeyFree      0x07000008 /* Key already free */
#define AN_BadChkSum    0x07000009 /* Invalid checksum */
#define AN_DiskError    0x0700000A /* Disk Error */
#define AN_KeyRange     0x0700000B /* Key out of range */
#define AN_BadOverlay   0x0700000C /* Bad overlay */
#define AN_BadInitFunc  0x0700000D /* Invalid init packet for cli/shell */
#define AN_FileReclosed 0x0700000E /* A filehandle was closed more than once */

/*------ ramlib.library */
#define AN_RAMLib       0x08000000
#define AN_BadSegList   0x08000001      /* no overlays in library seglists */

/*------ icon.library */
#define AN_IconLib      0x09000000

/*------ expansion.library */
#define AN_ExpansionLib 0x0A000000
#define AN_BadExpansionFree     0x0A000001 /* freeed free region */

/*------ diskfont.library */
#define AN_DiskfontLib  0x0B000000

/*------ audio.device */
#define AN_AudioDev     0x10000000

/*------ console.device */
#define AN_ConsoleDev   0x11000000
```

exec/alerts.h

Page 4

```
#define AN_NoWindow     0x11000001      /* Console can't open initial window */

/*------ gameport.device */
#define AN_GamePortDev  0x12000000

/*------ keyboard.device */
#define AN_KeyboardDev  0x13000000

/*------ trackdisk.device */
#define AN_TrackDiskDev 0x14000000
#define AN_TDCalibSeek  0x14000001      /* calibrate: seek error */
#define AN_TDDelay      0x14000002      /* delay: error on timer wait */

/*------ timer.device */
#define AN_TimerDev     0x15000000
#define AN_TMBadReq     0x15000001 /* bad request */
#define AN_TMBadSupply  0x15000002 /* power supply -- no 50/60Hz ticks */

/*------ cia.resource */
#define AN_CIARsrc      0x20000000

/*------ disk.resource */
#define AN_DiskRsrc     0x21000000
#define AN_DRHasDisk    0x21000001      /* get unit: already has disk */
#define AN_DRIntNoAct   0x21000002      /* interrupt: no active unit */

/*------ misc.resource */
#define AN_MiscRsrc     0x22000000

/*------ bootstrap */
#define AN_BootStrap    0x30000000
#define AN_BootError    0x30000001      /* boot code returned an error */

/*------ Workbench */
#define AN_Workbench                    0x31000000
#define AN_NoFonts                      0xB1000001
#define AN_WBBadStartupMsg1             0x31000001
#define AN_WBBadStartupMsg2             0x31000002
#define AN_WBBadIOMsg                   0x31000003

#define AN_WBInitPotionAllocDrawer      0xB1010004
#define AN_WBCreateWBMenusCreateMenus1  0xB1010005
#define AN_WBCreateWBMenusCreateMenus2  0xB1010006
#define AN_WBLayoutWBMenusLayoutMenus   0xB1010007
#define AN_WBAddToolMenuItem            0xB1010008
#define AN_WBReLayoutToolMenu           0xB1010009
#define AN_WBinitTimer                  0xB101000A
#define AN_WBInitLayerDemon             0xB101000B
#define AN_WBinitWbGels                 0xB101000C
#define AN_WBInitScreenAndWindows1      0xB101000D
#define AN_WBInitScreenAndWindows2      0xB101000E
#define AN_WBInitScreenAndWindows3      0xB101000F
#define AN_WBMAlloc                     0xB1010010

/*------ DiskCopy */
#define AN_DiskCopy     0x32000000

/*------ toolkit for Intuition */
#define AN_GadTools     0x33000000

/*------ System utility library */
#define AN_UtilityLib   0x34000000

/*------ For use by any application that needs it */
#define AN_Unknown      0x35000000

```

```
#endif /* EXEC_ALERTS_H */
```

exec/devices.h

```
#ifndef EXEC_DEVICES_H
#define EXEC_DEVICES_H
/*
**      $Filename: exec/devices.h $
**      $Release: 2.04 $
**      $Revision: 36.5 $
**      $Date: 90/05/10 $
**
**      Include file for use by Exec device drivers
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_LIBRARIES_H
#include "exec/libraries.h"
#endif /* EXEC_LIBRARIES_H */

#ifndef EXEC_PORTS_H
#include "exec/ports.h"
#endif /* EXEC_PORTS_H */


/****** Device ***********************************************************/

struct Device {
    struct  Library dd_Library;
};


/****** Unit *************************************************************/

struct Unit {
    struct  MsgPort unit_MsgPort;       /* queue for unprocessed messages */
                                        /* instance of msgport is recommended */
    UBYTE   unit_flags;
    UBYTE   unit_pad;
    UWORD   unit_OpenCnt;               /* number of active opens */
};


#define UNITF_ACTIVE    (1<<0)
#define UNITF_INTASK    (1<<1)

#endif  /* EXEC_DEVICES_H */
```

exec/errors.h

```
#ifndef EXEC_ERRORS_H
#define EXEC_ERRORS_H
/*
**      $Filename: exec/errors.h $
**      $Release: 2.04 $
**      $Revision: 36.6 $
**      $Date: 90/05/27 $
**
**      Standard Device IO Errors (returned in io_Error)
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#define IOERR_OPENFAIL    (-1) /* device/unit failed to open */
#define IOERR_ABORTED     (-2) /* request terminated early [after AbortIO()] */
#define IOERR_NOCMD       (-3) /* command not supported by device */
#define IOERR_BADLENGTH   (-4) /* not a valid length (usually IO_LENGTH) */
#define IOERR_BADADDRESS  (-5) /* invalid address (misaligned or bad range) */
#define IOERR_UNITBUSY    (-6) /* device opens ok, but requested unit is busy */
#define IOERR_SELFTEST    (-7) /* hardware failed self-test */

#endif  /* EXEC_ERRORS_H */
```

---

exec/exec.h

```
#ifndef EXEC_EXEC_H
#define EXEC_EXEC_H
/*
**      $Filename: exec/exec.h $
**      $Release: 2.04 $
**      $Revision: 36.5 $
**      $Date: 90/05/10 $
**
**      Include all other Exec include files in a non-overlapping order.
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#include "exec/types.h"
#include "exec/nodes.h"
#include "exec/lists.h"
#include "exec/alerts.h"
#include "exec/errors.h"
#include "exec/initializers.h"
#include "exec/resident.h"
#include "exec/memory.h"
#include "exec/tasks.h"
#include "exec/ports.h"
#include "exec/interrupts.h"
#include "exec/semaphores.h"
#include "exec/libraries.h"
#include "exec/io.h"
#include "exec/devices.h"
#include "exec/execbase.h"

#endif  /* EXEC_EXEC_H */
```

```
#ifndef EXEC_EXECBASE_H
#define EXEC_EXECBASE_H
/*
**      $Filename: exec/execbase.h $
**      $Release: 2.04 $
**      $Revision: 36.14 $
**      $Date: 91/02/19 $
**
**      Definition of the exec.library base structure.
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_LISTS_H
#include "exec/lists.h"
#endif /* EXEC_LISTS_H */

#ifndef EXEC_INTERRUPTS_H
#include "exec/interrupts.h"
#endif /* EXEC_INTERRUPTS_H */

#ifndef EXEC_LIBRARIES_H
#include "exec/libraries.h"
#endif /* EXEC_LIBRARIES_H */

#ifndef EXEC_TASKS_H
#include "exec/tasks.h"
#endif /* EXEC_TASKS_H */


/* Definition of the Exec library base structure (pointed to by location 4).
** Most fields are not to be viewed or modified by user programs.  Use
** extreme caution.
*/
struct ExecBase {
        struct Library LibNode; /* Standard library node */

/******** Static System Variables ********/

        UWORD   SoftVer;        /* kickstart release number (obs.) */
        WORD    LowMemChkSum;   /* checksum of 68000 trap vectors */
        ULONG   ChkBase;        /* system base pointer complement */
        APTR    ColdCapture;    /* coldstart soft capture vector */
        APTR    CoolCapture;    /* coolstart soft capture vector */
        APTR    WarmCapture;    /* warmstart soft capture vector */
        APTR    SysStkUpper;    /* system stack base   (upper bound) */
        APTR    SysStkLower;    /* top of system stack (lower bound) */
        ULONG   MaxLocMem;      /* top of chip memory */
        APTR    DebugEntry;     /* global debugger entry point */
        APTR    DebugData;      /* global debugger data segment */
        APTR    AlertData;      /* alert data segment */
        APTR    MaxExtMem;      /* top of extended mem, or null if none */

        UWORD   ChkSum; /* for all of the above (minus 2) */

/****** Interrupt Related ***************************************/

        struct  IntVector IntVects[16];

/****** Dynamic System Variables *************************************/

        struct  Task *ThisTask; /* pointer to current task (readable) */

        ULONG   IdleCount;      /* idle counter */
        ULONG   DispCount;      /* dispatch counter */
```

```
        UWORD   Quantum;        /* time slice quantum */
        UWORD   Elapsed;        /* current quantum ticks */
        UWORD   SysFlags;       /* misc internal system flags */
        BYTE    IDNestCnt;      /* interrupt disable nesting count */
        BYTE    TDNestCnt;      /* task disable nesting count */

        UWORD   AttnFlags;      /* special attention flags (readable) */

        UWORD   AttnResched;    /* rescheduling attention */
        APTR    ResModules;     /* resident module array pointer */
        APTR    TaskTrapCode;
        APTR    TaskExceptCode;
        APTR    TaskExitCode;
        ULONG   TaskSigAlloc;
        UWORD   TaskTrapAlloc;


/****** System Lists (private!) ********************************/

        struct  List MemList;
        struct  List ResourceList;
        struct  List DeviceList;
        struct  List IntrList;
        struct  List LibList;
        struct  List PortList;
        struct  List TaskReady;
        struct  List TaskWait;

        struct  SoftIntList SoftInts[5];

/****** Other Globals *******************************************/

        LONG    LastAlert[4];

        /* these next two variables are provided to allow
        ** system developers to have a rough idea of the
        ** period of two externally controlled signals --
        ** the time between vertical blank interrupts and the
        ** external line rate (which is counted by CIA A's
        ** "time of day" clock).  In general these values
        ** will be 50 or 60, and may or may not track each
        ** other.  These values replace the obsolete AFB_PAL
        ** and AFB_50HZ flags.
        */
        UBYTE   VBlankFrequency;        /* (readable) */
        UBYTE   PowerSupplyFrequency;   /* (readable) */

        struct  List SemaphoreList;

        /* these next two are to be able to kickstart into user ram.
        ** KickMemPtr holds a singly linked list of MemLists which
        ** will be removed from the memory list via AllocAbs.  If
        ** all the AllocAbs's succeeded, then the KickTagPtr will
        ** be added to the rom tag list.
        */
        APTR    KickMemPtr;     /* ptr to queue of mem lists */
        APTR    KickTagPtr;     /* ptr to rom tag queue */
        APTR    KickCheckSum;   /* checksum for mem and tags */

/****** V36 Exec additions start here **************************************/

        UWORD   ex_Pad0;
        ULONG   ex_Reserved0;
        APTR    ex_RamLibPrivate;
        /* The next ULONG contains the system "E" clock frequency,
        ** expressed in Hertz.  The E clock is used as a timebase for
```

exec/execbase.h

```
        ** the Amiga's 8520 I/O chips. (E is connected to "02").
        ** Typical values are 715909 for NTSC, or 709379 for PAL.
        */
        ULONG   ex_EClockFrequency;     /* (readable) */
        ULONG   ex_CacheControl;        /* Private to CacheControl calls */
        ULONG   ex_TaskID;              /* Next available task ID */

        ULONG   ex_PuddleSize;
        ULONG   ex_PoolThreshold;
        struct  MinList ex_PublicPool;

        APTR    ex_MMULock;             /* private */

        UBYTE   ex_Reserved[12];
};


/****** Bit defines for AttnFlags (see above) ******************************/

/*  Processors and Co-processors: */
#define AFB_68010       0       /* also set for 68020 */
#define AFB_68020       1       /* also set for 68030 */
#define AFB_68030       2       /* also set for 68040 */
#define AFB_68040       3
#define AFB_68881       4       /* also set for 68882 */
#define AFB_68882       5

#define AFF_68010       (1L<<0)
#define AFF_68020       (1L<<1)
#define AFF_68030       (1L<<2)
#define AFF_68040       (1L<<3)
#define AFF_68881       (1L<<4)
#define AFF_68882       (1L<<5)

/* #define AFB_RESERVED8   8 */
/* #define AFB_RESERVED9   9 */


/****** Selected flag definitions for Cache manipulation calls **********/

#define CACRF_EnableI       (1L<<0)  /* Enable instruction cache */
#define CACRF_FreezeI       (1L<<1)  /* Freeze instruction cache */
#define CACRF_ClearI        (1L<<3)  /* Clear instruction cache  */
#define CACRF_IBE           (1L<<4)  /* Instruction burst enable */
#define CACRF_EnableD       (1L<<8)  /* 68030 Enable data cache  */
#define CACRF_FreezeD       (1L<<9)  /* 68030 Freeze data cache  */
#define CACRF_ClearD        (1L<<11) /* 68030 Clear data cache   */
#define CACRF_DBE           (1L<<12) /* 68030 Data burst enable */
#define CACRF_WriteAllocate (1L<<13) /* 68030 Write-Allocate mode
                                        (must always be set!)    */
#define CACRF_CopyBack      (1L<<31) /* Master enable for copyback caches */

#endif  /* EXEC_EXECBASE_H */
```

exec/initializers.h

```
#ifndef EXEC_INITIALIZERS_H
#define EXEC_INITIALIZERS_H
/*
**      $Filename: exec/initializers.h $
**      $Release: 2.04 $
**      $Revision: 36.5 $
**      $Date: 90/05/10 $
**
**      Macros for use with the InitStruct() function.
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#define OFFSET(structName, structEntry) \
                                (&(((struct structName *) 0)->structEntry))
#define INITBYTE(offset,value)  0xe000,(UWORD) (offset),(UWORD) ((value)<<8)
#define INITWORD(offset,value)  0xd000,(UWORD) (offset),(UWORD) (value)
#define INITLONG(offset,value)  0xc000,(UWORD) (offset), \
                                (UWORD) ((value)>>16), \
                                (UWORD) ((value) & 0xffff)
#define INITSTRUCT(size,offset,value,count) \
                                (UWORD) (0xc000|(size<<12)|(count<<8)| \
                                ((UWORD) ((offset)>>16)), \
                                ((UWORD) (offset)) & 0xffff)
#endif /* EXEC_INITIALIZERS_H */
```

exec/interrupts.h

```
#ifndef EXEC_INTERRUPTS_H
#define EXEC_INTERRUPTS_H
/*
**      $Filename: exec/interrupts.h $
**      $Release: 2.04 $
**      $Revision: 36.4 $
**      $Date: 90/05/10 $
**
**      Callback structures used by hardware & software interrupts
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_NODES_H
#include "exec/nodes.h"
#endif /* EXEC_NODES_H */

#ifndef EXEC_LISTS_H
#include "exec/lists.h"
#endif /* EXEC_LISTS_H */


struct Interrupt {
    struct  Node is_Node;
    APTR    is_Data;                /* server data segment  */
    VOID    (*is_Code)();           /* server code entry    */
};


struct IntVector {              /* For EXEC use ONLY! */
    APTR    iv_Data;
    VOID    (*iv_Code)();
    struct  Node *iv_Node;
};


struct SoftIntList {            /* For EXEC use ONLY! */
    struct List sh_List;
    UWORD  sh_Pad;
};

#define SIH_PRIMASK (0xf0)

/* this is a fake INT definition, used only for AddIntServer and the like */
#define INTB_NMI        15
#define INTF_NMI        (1<<15)

#endif  /* EXEC_INTERRUPTS_H */
```

---

exec/io.h

```
#ifndef EXEC_IO_H
#define EXEC_IO_H
/*
**      $Filename: exec/io.h $
**      $Release: 2.04 $
**      $Revision: 36.4 $
**      $Date: 90/05/10 $
**
**      Message structures used for device communication
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_PORTS_H
#include "exec/ports.h"
#endif /* EXEC_PORTS_H */


struct IORequest {
    struct  Message io_Message;
    struct  Device  *io_Device;     /* device node pointer  */
    struct  Unit    *io_Unit;       /* unit (driver private)*/
    UWORD   io_Command;     /* device command */
    UBYTE   io_Flags;
    BYTE    io_Error;               /* error or warning num */
};

struct IOStdReq {
    struct  Message io_Message;
    struct  Device  *io_Device;     /* device node pointer  */
    struct  Unit    *io_Unit;       /* unit (driver private)*/
    UWORD   io_Command;     /* device command */
    UBYTE   io_Flags;
    BYTE    io_Error;               /* error or warning num */
    ULONG   io_Actual;              /* actual number of bytes transferred */
    ULONG   io_Length;              /* requested number bytes transferred*/
    APTR    io_Data;                /* points to data area */
    ULONG   io_Offset;              /* offset for block structured devices */
};

/* library vector offsets for device reserved vectors */
#define DEV_BEGINIO     (-30)
#define DEV_ABORTIO     (-36)

/* io_Flags defined bits */
#define IOB_QUICK       0
#define IOF_QUICK       (1<<0)


#define CMD_INVALID     0
#define CMD_RESET       1
#define CMD_READ        2
#define CMD_WRITE       3
#define CMD_UPDATE      4
#define CMD_CLEAR       5
#define CMD_STOP        6
#define CMD_START       7
#define CMD_FLUSH       8

#define CMD_NONSTD      9

#endif  /* EXEC_IO_H */
```

exec/libraries.h

Page 1

```
#ifndef EXEC_LIBRARIES_H
#define EXEC_LIBRARIES_H
/*
**      $Filename: exec/libraries.h $
**      $Release: 2.04 $
**      $Revision: 36.10 $
**      $Date: 90/05/10 $
**
**      Definitions for use when creating or using Exec libraries
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_NODES_H
#include "exec/nodes.h"
#endif /* EXEC_NODES_H */

/*------ Special Constants ---------------------------------------*/
#define LIB_VECTSIZE    6       /* Each library entry takes 6 bytes */
#define LIB_RESERVED    4       /* Exec reserves the first 4 vectors */
#define LIB_BASE        (-LIB_VECTSIZE)
#define LIB_USERDEF     (LIB_BASE-(LIB_RESERVED*LIB_VECTSIZE))
#define LIB_NONSTD      (LIB_USERDEF)

/*------ Standard Functions ---------------------------------------*/
#define LIB_OPEN        (-6)
#define LIB_CLOSE       (-12)
#define LIB_EXPUNGE     (-18)
#define LIB_EXTFUNC     (-24)   /* for future expansion */

/*------ Library Base Structure ----------------------------------*/
/* Also used for Devices and some Resources */
struct Library {
    struct  Node lib_Node;
    UBYTE   lib_Flags;
    UBYTE   lib_pad;
    UWORD   lib_NegSize;            /* number of bytes before library */
    UWORD   lib_PosSize;            /* number of bytes after library */
    UWORD   lib_Version;            /* major */
    UWORD   lib_Revision;           /* minor */
    APTR    lib_IdString;           /* ASCII identification */
    ULONG   lib_Sum;                /* the checksum itself */
    UWORD   lib_OpenCnt;            /* number of current opens */
};      /* Warning: size is not a longword multiple! */

/* lib_Flags bit definitions (all others are system reserved) */
#define LIBF_SUMMING    (1<<0)      /* we are currently checksumming */
#define LIBF_CHANGED    (1<<1)      /* we have just changed the lib */
#define LIBF_SUMUSED    (1<<2)      /* set if we should bother to sum */
#define LIBF_DELEXP     (1<<3)      /* delayed expunge */

/* Temporary Compatibility */
#define lh_Node lib_Node
#define lh_Flags        lib_Flags
#define lh_pad          lib_pad
#define lh_NegSize      lib_NegSize
#define lh_PosSize      lib_PosSize
#define lh_Version      lib_Version
#define lh_Revision     lib_Revision
#define lh_IdString     lib_IdString
#define lh_Sum          lib_Sum
#define lh_OpenCnt      lib_OpenCnt

#endif  /* EXEC_LIBRARIES_H */
```

exec/lists.h

Page 1

```
#ifndef EXEC_LISTS_H
#define EXEC_LISTS_H
/*
**      $Filename: exec/lists.h $
**      $Release: 2.04 $
**      $Revision: 36.9 $
**      $Date: 91/02/19 $
**
**      Definitions and macros for use with Exec lists
**
**      (C) Copyright 1985,1986,1987,1988,1989,1991 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_NODES_H
#include "exec/nodes.h"
#endif /* EXEC_NODES_H */

/*
 *  Full featured list header.
 */
struct List {
   struct  Node *lh_Head;
   struct  Node *lh_Tail;
   struct  Node *lh_TailPred;
   UBYTE   lh_Type;
   UBYTE   l_pad;
};      /* word aligned */

/*
 * Minimal List Header - no type checking
 */
struct MinList {
   struct  MinNode *mlh_Head;
   struct  MinNode *mlh_Tail;
   struct  MinNode *mlh_TailPred;
};      /* longword aligned */


/*
 *      Check for the presence of any nodes on the given list.  These
 *      macros are even safe to use on lists that are modified by other
 *      tasks.  However; if something is simultaneously changing the
 *      list, the result of the test is unpredictable.
 *
 *      Unless you first arbitrated for ownership of the list, you can't
 *      _depend_ on the contents of the list.  Nodes might have been added
 *      or removed during or after the macro executes.
 *
 *              if( IsListEmpty(list) )         printf("List is empty\n");
 */
#define IsListEmpty(x) \
        ( ((x)->lh_TailPred) == (struct Node *)(x) )

#define IsMsgPortEmpty(x) \
        ( ((x)->mp_MsgList.lh_TailPred) == (struct Node *)(&(x)->mp_MsgList) )


#endif  /* EXEC_LISTS_H */
```

exec/memory.h

Page 1

```
#ifndef EXEC_MEMORY_H
#define EXEC_MEMORY_H
/*
**      $Filename: exec/memory.h $
**      $Release: 2.04 $
**      $Revision: 36.11 $
**      $Date: 90/06/11 $
**
**      Definitions and structures used by the memory allocation system
**
**      (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_NODES_H
#include "exec/nodes.h"
#endif /* EXEC_NODES_H */


/****** MemChunk ****************************************************/

struct  MemChunk {
    struct  MemChunk *mc_Next;  /* pointer to next chunk */
    ULONG   mc_Bytes;           /* chunk byte size      */
};


/****** MemHeader ***************************************************/

struct  MemHeader {
    struct  Node mh_Node;
    UWORD   mh_Attributes;      /* characteristics of this region */
    struct  MemChunk *mh_First; /* first free region            */
    APTR    mh_Lower;           /* lower memory bound           */
    APTR    mh_Upper;           /* upper memory bound+1 */
    ULONG   mh_Free;            /* total number of free bytes   */
};


/****** MemEntry ****************************************************/

struct  MemEntry {
union {
    ULONG   meu_Reqs;           /* the AllocMem requirements */
    APTR    meu_Addr;           /* the address of this memory region */
    } me_Un;
    ULONG   me_Length;          /* the length of this memory region */
};

#define me_un       me_Un       /* compatibility - do not use*/
#define me_Reqs     me_Un.meu_Reqs
#define me_Addr     me_Un.meu_Addr


/****** MemList *****************************************************/

/* Note: sizeof(struct MemList) includes the size of the first MemEntry! */
struct  MemList {
    struct  Node ml_Node;
    UWORD   ml_NumEntries;      /* number of entries in this struct */
    struct  MemEntry ml_ME[1];  /* the first entry      */
};

#define ml_me   ml_ME           /* compatability - do not use */


```

exec/memory.h

Page 2

```
/*----- Memory Requirement Types ---------------------------*/
/*----- See the AllocMem() documentation for details--------*/

#define MEMF_ANY    (0L)        /* Any type of memory will do */
#define MEMF_PUBLIC (1L<<0)
#define MEMF_CHIP   (1L<<1)
#define MEMF_FAST   (1L<<2)
#define MEMF_LOCAL  (1L<<8)
#define MEMF_24BITDMA (1L<<9)   /* DMAable memory within 24 bits of address */

#define MEMF_CLEAR   (1L<<16)
#define MEMF_LARGEST (1L<<17)
#define MEMF_REVERSE (1L<<18)
#define MEMF_TOTAL   (1L<<19)   /* AvailMem: return total size of memory */

#define MEM_BLOCKSIZE   8L
#define MEM_BLOCKMASK   (MEM_BLOCKSIZE-1)

#endif  /* EXEC_MEMORY_H */
```

exec/nodes.h

Page 1

```
#ifndef EXEC_NODES_H
#define EXEC_NODES_H
/*
**      $Filename: exec/nodes.h $
**      $Release: 2.04 $
**      $Revision: 36.11 $
**      $Date: 91/01/09 $
**
**      Nodes & Node type identifiers.
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_TYPES_H
#include "exec/types.h"
#endif /* EXEC_TYPES_H */


/*
 *  List Node Structure.  Each member in a list starts with a Node
 */

struct Node {
    struct  Node *ln_Succ;      /* Pointer to next (successor) */
    struct  Node *ln_Pred;      /* Pointer to previous (predecessor) */
    UBYTE   ln_Type;
    BYTE    ln_Pri;             /* Priority, for sorting */
    char    *ln_Name;           /* ID string, null terminated */
};      /* Note: word aligned */

/* minimal node -- no type checking possible */
struct MinNode {
    struct MinNode *mln_Succ;
    struct MinNode *mln_Pred;
};


/*
** Note: Newly initialized IORequests, and software interrupt structures
** used with Cause(), should have type NT_UNKNOWN.  The OS will assign a type
** when they are first used.
*/
/*----- Node Types for LN_TYPE -----*/
#define NT_UNKNOWN      0
#define NT_TASK         1       /* Exec task */
#define NT_INTERRUPT    2
#define NT_DEVICE       3
#define NT_MSGPORT      4
#define NT_MESSAGE      5       /* Indicates message currently pending */
#define NT_FREEMSG      6
#define NT_REPLYMSG     7       /* Message has been replied */
#define NT_RESOURCE     8
#define NT_LIBRARY      9
#define NT_MEMORY       10
#define NT_SOFTINT      11      /* Internal flag used by SoftInits */
#define NT_FONT         12
#define NT_PROCESS      13      /* AmigaDOS Process */
#define NT_SEMAPHORE    14
#define NT_SIGNALSEM    15      /* signal semaphores */
#define NT_BOOTNODE     16
#define NT_KICKMEM      17
#define NT_GRAPHICS     18
#define NT_DEATHMESSAGE 19

#define NT_USER         254     /* User node types work down from here */
```

exec/nodes.h

Page 2

```
#define NT_EXTENDED     255

#endif  /* EXEC_NODES_H */
```

exec/ports.h

```c
#ifndef EXEC_PORTS_H
#define EXEC_PORTS_H
/*
**      $Filename: exec/ports.h $
**      $Release: 2.04 $
**      $Revision: 36.5 $
**      $Date: 90/05/10 $
**
**      Message ports and Messages.
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_NODES_H
#include "exec/nodes.h"
#endif /* EXEC_NODES_H */

#ifndef EXEC_LISTS_H
#include "exec/lists.h"
#endif /* EXEC_LISTS_H */

#ifndef EXEC_TASKS_H
#include "exec/tasks.h"
#endif /* EXEC_TASKS_H */


/****** MsgPort *****************************************************/

struct MsgPort {
    struct  Node mp_Node;
    UBYTE   mp_Flags;
    UBYTE   mp_SigBit;          /* signal bit number    */
    void   *mp_SigTask;         /* object to be signalled */
    struct  List mp_MsgList;    /* message linked list  */
};

#define mp_SoftInt mp_SigTask   /* Alias */

/* mp_Flags: Port arrival actions (PutMsg) */
#define PF_ACTION       3       /* Mask */
#define PA_SIGNAL       0       /* Signal task in mp_SigTask */
#define PA_SOFTINT      1       /* Signal SoftInt in mp_SoftInt/mp_SigTask */
#define PA_IGNORE       2       /* Ignore arrival */


/****** Message *****************************************************/

struct Message {
    struct  Node mn_Node;
    struct  MsgPort *mn_ReplyPort;  /* message reply port */
    UWORD   mn_Length;              /* total message length, in bytes */
                                    /* (include the size of the Message */
                                    /* structure in the length) */
};

#endif  /* EXEC_PORTS_H */
```

---

exec/resident.h

```c
#ifndef EXEC_RESIDENT_H
#define EXEC_RESIDENT_H
/*
**      $Filename: exec/resident.h $
**      $Release: 2.04 $
**      $Revision: 36.8 $
**      $Date: 90/11/01 $
**
**      Resident/ROMTag stuff.  Used to identify and initialize code modules.
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_TYPES_H
#include "exec/types.h"
#endif /* EXEC_TYPES_H */


struct Resident {
    UWORD rt_MatchWord; /* word to match on (ILLEGAL)    */
    struct Resident *rt_MatchTag; /* pointer to the above       */
    APTR  rt_EndSkip;             /* address to continue scan   */
    UBYTE rt_Flags;               /* various tag flags          */
    UBYTE rt_Version;             /* release version number     */
    UBYTE rt_Type;                /* type of module (NT_XXXXXX) */
    BYTE  rt_Pri;                 /* initialization priority */
    char  *rt_Name;               /* pointer to node name */
    char  *rt_IdString; /* pointer to identification string */
    APTR  rt_Init;                /* pointer to init code */
};

#define RTC_MATCHWORD   0x4AFC  /* The 68000 "ILLEGAL" instruction */

#define RTF_AUTOINIT    (1<<7)  /* rt_Init points to data structure */
#define RTF_AFTERDOS    (1<<2)
#define RTF_SINGLETASK  (1<<1)
#define RTF_COLDSTART   (1<<0)

/* Compatibility: (obsolete) */
/* #define RTM_WHEN        3 */
#define RTW_NEVER       0
#define RTW_COLDSTART   1

#endif  /* EXEC_RESIDENT_H */
```

exec/semaphores.h

```
#ifndef EXEC_SEMAPHORES_H
#define EXEC_SEMAPHORES_H
/*
**      $Filename: exec/semaphores.h $
**      $Release: 2.04 $
**      $Revision: 36.6 $
**      $Date: 90/05/10 $
**
**      Definitions for locking functions.
**
**      (C) Copyright 1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_NODES_H
#include "exec/nodes.h"
#endif /* EXEC_NODES_H */

#ifndef EXEC_LISTS_H
#include "exec/lists.h"
#endif /* EXEC_LISTS_H */

#ifndef EXEC_PORTS_H
#include "exec/ports.h"
#endif /* EXEC_PORTS_H */

#ifndef EXEC_TASKS_H
#include "exec/tasks.h"
#endif /* EXEC_TASKS_H */


/****** SignalSemaphore **********************************************/

/* Private structure used by ObtainSemaphore() */
struct SemaphoreRequest {
    struct  MinNode sr_Link;
    struct  Task *sr_Waiter;
};

/* Signal Semaphore data structure */
struct SignalSemaphore {
    struct  Node ss_Link;
    WORD    ss_NestCount;
    struct  MinList ss_WaitQueue;
    struct  SemaphoreRequest ss_MultipleLink;
    struct  Task *ss_Owner;
    WORD    ss_QueueCount;
};


/****** Semaphore (Procure/Vacate type) ******************************/

struct Semaphore {
    struct MsgPort sm_MsgPort;
    WORD   sm_Bids;
};

#define sm_LockMsg      mp_SigTask


#endif  /* EXEC_SEMAPHORES_H */
```

exec/tasks.h

```
#ifndef EXEC_TASKS_H
#define EXEC_TASKS_H
/*
**      $Filename: exec/tasks.h $
**      $Release: 2.04 $
**      $Revision: 36.8 $
**      $Date: 90/05/30 $
**
**      Task Control Block, Singals, and Task flags.
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_NODES_H
#include "exec/nodes.h"
#endif /* EXEC_NODES_H */

#ifndef EXEC_LISTS_H
#include "exec/lists.h"
#endif /* EXEC_LISTS_H */


/* Please use Exec functions to modify task structure fields, where available.
 */
struct Task {
    struct  Node tc_Node;
    UBYTE   tc_Flags;
    UBYTE   tc_State;
    BYTE    tc_IDNestCnt;           /* intr disabled nesting*/
    BYTE    tc_TDNestCnt;           /* task disabled nesting*/
    ULONG   tc_SigAlloc;            /* sigs allocated */
    ULONG   tc_SigWait;     /* sigs we are waiting for */
    ULONG   tc_SigRecvd;            /* sigs we have received */
    ULONG   tc_SigExcept;           /* sigs we will take excepts for */
    UWORD   tc_TrapAlloc;           /* traps allocated */
    UWORD   tc_TrapAble;            /* traps enabled */
    APTR    tc_ExceptData;          /* points to except data */
    APTR    tc_ExceptCode;          /* points to except code */
    APTR    tc_TrapData;            /* points to trap code */
    APTR    tc_TrapCode;            /* points to trap data */
    APTR    tc_SPReg;               /* stack pointer        */
    APTR    tc_SPLower;     /* stack lower bound    */
    APTR    tc_SPUpper;     /* stack upper bound + 2*/
    VOID    (*tc_Switch)();         /* task losing CPU    */
    VOID    (*tc_Launch)();         /* task getting CPU  */
    struct  List tc_MemEntry;       /* Allocated memory. Freed by RemTask() */
    APTR    tc_UserData;            /* For use by the task; no restrictions! */
};

/*----- Flag Bits ------------------------------------------*/
#define TB_PROCTIME     0
#define TB_ETASK        3
#define TB_STACKCHK     4
#define TB_EXCEPT       5
#define TB_SWITCH       6
#define TB_LAUNCH       7

#define TF_PROCTIME     (1<<0)
#define TF_ETASK        (1<<3)
#define TF_STACKCHK     (1<<4)
#define TF_EXCEPT       (1<<5)
#define TF_SWITCH       (1<<6)
#define TF_LAUNCH       (1<<7)

/*----- Task States ----------------------------------------*/
```

exec/tasks.h

```
#define TS_INVALID      0
#define TS_ADDED        1
#define TS_RUN          2
#define TS_READY        3
#define TS_WAIT 4
#define TS_EXCEPT       5
#define TS_REMOVED      6

/*----- Predefined Signals -------------------------------------*/
#define SIGB_ABORT      0
#define SIGB_CHILD      1
#define SIGB_BLIT       4       /* Note: same as SINGLE */
#define SIGB_SINGLE     4       /* Note: same as BLIT */
#define SIGB_INTUITION  5
#define SIGB_DOS        8

#define SIGF_ABORT      (1L<<0)
#define SIGF_CHILD      (1L<<1)
#define SIGF_BLIT       (1L<<4)
#define SIGF_SINGLE     (1L<<4)
#define SIGF_INTUITION  (1L<<5)
#define SIGF_DOS        (1L<<8)

#endif  /* EXEC_TASKS_H */
```

exec/types.h

```
#ifndef EXEC_TYPES_H
#define EXEC_TYPES_H
/*
**      $Filename: exec/types.h $
**      $Release: 2.04 $
**      $Revision: 36.12 $
**      $Date: 91/02/15 $
**
**      Data typing.  Must be included before any other Amiga include.
**
**      (C) Copyright 1985,1986,1987,1988,1989,1990,1991 Commodore-Amiga, Inc.
**          All Rights Reserved
*/


#define INCLUDE_VERSION 36 /* Version of the include files in use. (Do not
                              use this label for OpenLibrary() calls!) */


#define GLOBAL  extern      /* the declaratory use of an external */
#define IMPORT  extern      /* reference to an external */
#define STATIC  static      /* a local static variable */
#define REGISTER register   /* a (hopefully) register variable */


#ifndef VOID
#define VOID            void
#endif


 /*  WARNING: APTR was redefined for the V36 Includes!  APTR is a   */
 /*  32-Bit Absolute Memory Pointer.  C pointer math will not      */
/*  operate on APTR --  use "ULONG *" instead.                   */
#ifndef APTR_TYPEDEF
#define APTR_TYPEDEF
typedef void            *APTR;      /* 32-bit untyped pointer */
#endif
typedef long            LONG;       /* signed 32-bit quantity */
typedef unsigned long   ULONG;      /* unsigned 32-bit quantity */
typedef unsigned long   LONGBITS;   /* 32 bits manipulated individually */
typedef short           WORD;       /* signed 16-bit quantity */
typedef unsigned short  UWORD;      /* unsigned 16-bit quantity */
typedef unsigned short  WORDBITS;   /* 16 bits manipulated individually */
#ifdef __STDC__
typedef signed char     BYTE;       /* signed 8-bit quantity */
#else
typedef char            BYTE;       /* signed 8-bit quantity */
#endif
typedef unsigned char   UBYTE;      /* unsigned 8-bit quantity */
typedef unsigned char   BYTEBITS;   /* 8 bits manipulated individually */
typedef short           RPTR;       /* signed relative pointer */
typedef unsigned char  *STRPTR;     /* string pointer (NULL terminated) */


/* For compatibility only: (don't use in new code) */
typedef short           SHORT;      /* signed 16-bit quantity (use WORD) */
typedef unsigned short  USHORT;     /* unsigned 16-bit quantity (use UWORD) */
typedef short           COUNT;
typedef unsigned short  UCOUNT;
typedef ULONG           CPTR;


/* Types with specific semantics */
typedef float           FLOAT;
typedef double          DOUBLE;
typedef short           BOOL;
```

exec/types.h

```
typedef unsigned char   TEXT;

#ifndef TRUE
#define TRUE            1
#endif
#ifndef FALSE
#define FALSE           0
#endif
#ifndef NULL
#define NULL            0L
#endif


#define BYTEMASK        0xFF


 /* #define LIBRARY_VERSION is now obsolete.  Please use LIBRARY_MINIMUM */
/* or code the specific minimum library version you require.            */
#define LIBRARY_MINIMUM 33 /* Lowest version supported by Commodore-Amiga */


#endif  /* EXEC_TYPES_H */
```

graphics/clip.h Page 1

```
#ifndef GRAPHICS_CLIP_H
#define GRAPHICS_CLIP_H
/*
**      $Filename: graphics/clip.h $
**      $Release: 2.04 $
**      $Revision: 37.1 $
**      $Date: 91/01/28 $
**
**
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_TYPES_H
#include <exec/types.h>
#endif

#ifndef GRAPHICS_GFX_H
#include <graphics/gfx.h>
#endif
#ifndef EXEC_SEMAPHORES_H
#include <exec/semaphores.h>
#endif
#ifndef UTILITY_HOOKS_H
#include <utility/hooks.h>
#endif

/* structures used by and constructed by windowlib.a */
/* understood by rom software */

#define NEWLOCKS

struct Layer
{
    struct  Layer *front,*back;
    struct  ClipRect    *ClipRect;  /* read by roms to find first cliprect */
    struct  RastPort    *rp;
    struct  Rectangle   bounds;
    UBYTE   reserved[4];
    UWORD   priority;               /* system use only */
    UWORD   Flags;                  /* obscured ?, Virtual BitMap? */
    struct  BitMap *SuperBitMap;
    struct  ClipRect *SuperClipRect; /* super bitmap cliprects if VBitMap != 0*/
                                  /* else damage cliprect list for refresh */
    APTR    Window;               /* reserved for user interface use */
    WORD   Scroll_X,Scroll_Y;
    struct  ClipRect *cr,*cr2,*crnew;   /* used by dedice */
    struct  ClipRect *SuperSaveClipRects; /* preallocated cr's */
    struct  ClipRect *_cliprects;       /* system use during refresh */
    struct  Layer_Info  *LayerInfo;     /* points to head of the list */
    struct  SignalSemaphore Lock;
    struct  Hook *BackFill;
    ULONG   reserved1;
    struct  Region *ClipRegion;
    struct  Region *saveClipRects;      /* used to back out when in trouble*/
    WORD    Width,Height;               /* system use */
    UBYTE   reserved2[18];
    /* this must stay here */
    struct  Region  *DamageList;    /* list of rectangles to refresh
                                       through */
};

struct ClipRect
{
    struct  ClipRect *Next;         /* roms used to find next ClipRect */
```

graphics/clip.h Page 2

```
    struct  ClipRect *prev;         /* ignored by roms, used by windowlib */
    struct  Layer   *lobs;          /* ignored by roms, used by windowlib */
    struct  BitMap  *BitMap;
    struct  Rectangle   bounds;     /* set up by windowlib, used by roms */
    struct  ClipRect *_p1,*_p2;     /* system reserved */
    LONG    reserved;               /* system use */
#ifdef NEWCLIPRECTS_1_1
    LONG    Flags;                  /* only exists in layer allocation */
#endif                              /* MUST be multiple of 8 bytes to buffer */
};

/* internal cliprect flags */
#define CR_NEEDS_NO_CONCEALED_RASTERS  1
#define CR_NEEDS_NO_LAYERBLIT_DAMAGE   2

/* defines for code values for getcode */
#define ISLESSX 1
#define ISLESSY 2
#define ISGRTRX 4
#define ISGRTRY 8

#endif  /* GRAPHICS_CLIP_H */
```

graphics/collide.h

Page 1

```
#ifndef GRAPHICS_COLLIDE_H
#define GRAPHICS_COLLIDE_H
/*
**      $Filename: graphics/collide.h $
**      $Release: 2.04 $
**      $Revision: 37.0 $
**      $Date: 91/01/07 $
**
**      include file for collision detection and control
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

/* These bit descriptors are used by the GEL collide routines.
 *  These bits are set in the hitMask and meMask variables of
 *  a GEL to describe whether or not these types of collisions
 *  can affect the GEL.  BNDRY_HIT is described further below;
 *  this bit is permanently assigned as the boundary-hit flag.
 *  The other bit GEL_HIT is meant only as a default to cover
 *  any GEL hitting any other; the user may redefine this bit.
 */
#define BORDERHIT 0

/* These bit descriptors are used by the GEL boundry hit routines.
 *  When the user's boundry-hit routine is called (via the argument
 *  set by a call to SetCollision) the first argument passed to
 *  the user's routine is the address of the GEL involved in the
 *  boundry-hit, and the second argument has the appropriate bit(s)
 *  set to describe which boundry was surpassed
 */
#define TOPHIT    1
#define BOTTOMHIT 2
#define LEFTHIT   4
#define RIGHTHIT  8

#endif  /* GRAPHICS_COLLIDE_H */
```

graphics/copper.h

Page 1

```
#ifndef GRAPHICS_COPPER_H
#define GRAPHICS_COPPER_H
/*
**      $Filename: graphics/copper.h $
**      $Release: 2.04 $
**      $Revision: 37.0 $
**      $Date: 91/01/07 $
**
**      graphics copper list intstruction definintions
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_TYPES_H
#include <exec/types.h>
#endif

#define COPPER_MOVE 0       /* pseude opcode for move #XXXX,dir */
#define COPPER_WAIT 1       /* pseudo opcode for wait y,x */
#define CPRNXTBUF   2       /* continue processing with next buffer */
#define CPR_NT_LOF  0x8000  /* copper instruction only for short frames */
#define CPR_NT_SHT  0x4000  /* copper instruction only for long frames */
#define CPR_NT_SYS  0x2000  /* copper user instruction only */

struct CopIns
{
    WORD   OpCode; /* 0 = move, 1 = wait */
    union
    {
    struct CopList *nxtlist;
    struct
    {
   union
   {
   WORD   VWaitPos;        /* vertical beam wait */
   WORD   DestAddr;        /* destination address of copper move */
   } u1;
   union
   {
   WORD   HWaitPos;        /* horizontal beam wait position */
   WORD   DestData;        /* destination immediate data to send */
   } u2;
    } u4;
    } u3;
};
/* shorthand for above */
#define NXTLIST     u3.nxtlist
#define VWAITPOS    u3.u4.u1.VWaitPos
#define DESTADDR    u3.u4.u1.DestAddr
#define HWAITPOS    u3.u4.u2.HWaitPos
#define DESTDATA    u3.u4.u2.DestData


/* structure of cprlist that points to list that hardware actually executes */
struct cprlist
{
    struct cprlist *Next;
    UWORD   *start;         /* start of copper list */
    WORD   MaxCount;       /* number of long instructions */
};

struct CopList
{
    struct  CopList *Next;  /* next block for this copper list */
    struct  CopList *_CopList;  /* system use */
```

graphics/copper.h

```
    struct  ViewPort *_ViewPort;    /* system use */
    struct  CopIns *CopIns; /* start of this block */
    struct  CopIns *CopPtr; /* intermediate ptr */
    UWORD   *CopLStart;     /* mrgcop fills this in for Long Frame*/
    UWORD   *CopSStart;     /* mrgcop fills this in for Short Frame*/
    WORD   Count;          /* intermediate counter */
    WORD   MaxCount;       /* max # of copins for this block */
    WORD   DyOffset;       /* offset this copper list vertical waits */
#ifdef V1_3
    UWORD   *Cop2Start;
    UWORD   *Cop3Start;
    UWORD   *Cop4Start;
    UWORD   *Cop5Start;
#endif
};

struct UCopList
{
    struct UCopList *Next;
    struct CopList  *FirstCopList; /* head node of this copper list */
    struct CopList  *CopList;      /* node in use */
};

/* private graphics data structure */

struct copinit
{
    UWORD vsync_hblank[2];
    UWORD diwstart[4];
    UWORD diagstrt[4];      /* copper list for first bitplane */
    UWORD sprstrtup[(2*8*2)];
    UWORD wait14[2];
    UWORD norm_hblank[2];
    UWORD genloc[4];
    UWORD jump[(2*2)];
    UWORD wait_forever[2];
    UWORD   sprstop[4];
};

#endif  /* GRAPHICS_COPPER_H */
```

graphics/display.h

```
#ifndef GRAPHICS_DISPLAY_H
#define GRAPHICS_DISPLAY_H
/*
**      $Filename: graphics/display.h $
**      $Release: 2.04 $
**      $Revision: 37.0 $
**      $Date: 91/01/07 $
**
**      include define file for display control registers
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

/* bplcon0 defines */
#define MODE_640    0x8000
#define PLNCNTMSK   0x7     /* how many bit planes? */
                                        /* 0 = none, 1->6 = 1->6, 7 = reserved */
#define PLNCNTSHFT  12                  /* bits to shift for bplcon0 */
#define PF2PRI      0x40                /* bplcon2 bit */
#define COLORON     0x0200              /* disable color burst */
#define DBLPF       0x400
#define HOLDNMODIFY 0x800
#define INTERLACE   4                   /* interlace mode for 400 */

/* bplcon1 defines */
#define PFA_FINE_SCROLL       0xF
#define PFB_FINE_SCROLL_SHIFT 4
#define PF_FINE_SCROLL_MASK   0xF

/* display window start and stop defines */
#define DIW_HORIZ_POS   0x7F       /* horizontal start/stop */
#define DIW_VRTCL_POS   0x1FF      /* vertical start/stop */
#define DIW_VRTCL_POS_SHIFT 7

/* Data fetch start/stop horizontal position */
#define DFTCH_MASK      0xFF

/* vposr bits */
#define VPOSRLOF        0x8000

#endif  /* GRAPHICS_DISPLAY_H */
```

graphics/displayinfo.h

```
#ifndef GRAPHICS_DISPLAYINFO_H
#define GRAPHICS_DISPLAYINFO_H
/*
**      $Filename: graphics/displayinfo.h $
**      $Release: 2.04 $
**      $Revision: 37.3 $
**      $Date: 91/01/14 $
**
**      include define file for displayinfo database
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_TYPES_H
#include <exec/types.h>
#endif /* EXEC_TYPES_H */

#ifndef GRAPHICS_GFX_H
#include <graphics/gfx.h>
#endif /* GRAPHICS_GFX_H */

#ifndef GRAPHICS_MONITOR_H
#include <graphics/monitor.h>
#endif /* GRAPHICS_MONITOR_H */

#ifndef UTILITY_TAGITEM_H
#include <utility/tagitem.h>
#endif /* UTILITY_TAGITEM_H */

/* the "public" handle to a DisplayInfoRecord */

typedef APTR DisplayInfoHandle;

/* datachunk type identifiers */

#define DTAG_DISP                   0x80000000
#define DTAG_DIMS                   0x80001000
#define DTAG_MNTR                   0x80002000
#define DTAG_NAME                   0x80003000

struct QueryHeader
{
        ULONG   StructID;       /* datachunk type identifier */
        ULONG   DisplayID;      /* copy of display record key   */
        ULONG   SkipID;         /* TAG_SKIP -- see tagitems.h */
        ULONG   Length;         /* length of local data in double-longwords */
};

struct DisplayInfo
{
        struct  QueryHeader Header;
        UWORD   NotAvailable;   /* if NULL available, else see defines */
        ULONG   PropertyFlags;  /* Properties of this mode see defines */
        Point   Resolution;     /* ticks-per-pixel X/Y                 */
        UWORD   PixelSpeed;     /* aproximation in nanoseconds         */
        UWORD   NumStdSprites;  /* number of standard amiga sprites    */
        UWORD   PaletteRange;   /* distinguishable shades available    */
        Point   SpriteResolution; /* std sprite ticks-per-pixel X/Y    */
        UBYTE   pad[4];
        ULONG   reserved[2];    /* terminator */
};

/* availability */

#define DI_AVAIL_NOCHIPS        0x0001
```

```
#define DI_AVAIL_NOMONITOR      0x0002
#define DI_AVAIL_NOTWITHGENLOCK 0x0004

/* mode properties */

#define DIPF_IS_LACE            0x00000001
#define DIPF_IS_DUALPF          0x00000002
#define DIPF_IS_PF2PRI          0x00000004
#define DIPF_IS_HAM             0x00000008

#define DIPF_IS_ECS             0x00000010      /*      note: ECS modes (SHIRES,
    VGA, and **
    **      PRODUCTIVITY) do not support      **
    **      attached sprites.                 **
    */
#define DIPF_IS_PAL             0x00000020
#define DIPF_IS_SPRITES 0x00000040
#define DIPF_IS_GENLOCK 0x00000080

#define DIPF_IS_WB              0x00000100
#define DIPF_IS_DRAGGABLE       0x00000200
#define DIPF_IS_PANELLED        0x00000400
#define DIPF_IS_BEAMSYNC        0x00000800

#define DIPF_IS_EXTRAHALFBRITE  0x00001000

struct DimensionInfo
{
        struct  QueryHeader Header;
        UWORD   MaxDepth;               /* log2( max number of colors ) */
        UWORD   MinRasterWidth;         /* minimum width in pixels      */
        UWORD   MinRasterHeight;        /* minimum height in pixels     */
        UWORD   MaxRasterWidth;         /* maximum width in pixels      */
        UWORD   MaxRasterHeight;        /* maximum height in pixels     */
        struct  Rectangle   Nominal;    /* "standard" dimensions        */
        struct  Rectangle   MaxOScan;   /* fixed, hardware dependant    */
        struct  Rectangle VideoOScan;   /* fixed, hardware dependant    */
        struct  Rectangle   TxtOScan;   /* editable via preferences     */
        struct  Rectangle   StdOScan;   /* editable via preferences     */
        UBYTE   pad[14];
        ULONG   reserved[2];            /* terminator */
};

struct MonitorInfo
{
        struct  QueryHeader Header;
        struct  MonitorSpec  *Mspc;     /* pointer to monitor specification  */
        Point   ViewPosition;           /* editable via preferences          */
        Point   ViewResolution;         /* standard monitor ticks-per-pixel  */
        struct  Rectangle ViewPositionRange;  /* fixed, hardware dependant */
        UWORD   TotalRows;              /* display height in scanlines       */
        UWORD   TotalColorClocks;       /* scanline width in 280 ns units    */
        UWORD   MinRow;         /* absolute minimum active scanline  */
        WORD    Compatibility;          /* how this coexists with others     */
        UBYTE   pad[36];
        ULONG   reserved[2];            /* terminator */
};

/* monitor compatibility */

#define MCOMPAT_MIXED   0       /* can share display with other MCOMPAT_MIXED */
#define MCOMPAT_SELF    1       /* can share only within same monitor *7
#define MCOMPAT_NOBODY -1       /* only one viewport at a time */
```

```

#define DISPLAYNAMELEN 32

struct NameInfo
{
        struct  QueryHeader Header;
        UBYTE   Name[DISPLAYNAMELEN];
        ULONG   reserved[2];          /* terminator */
};

/* DisplayInfoRecord identifiers */

#define INVALID_ID              ~0

/* normal identifiers */

#define MONITOR_ID_MASK         0xFFFF1000

#define DEFAULT_MONITOR_ID              0x00000000
#define NTSC_MONITOR_ID         0x00011000
#define PAL_MONITOR_ID                  0x00021000

/* the following 20 composite keys are for Modes on the default Monitor */
/* ntsc & pal "flavors" of these particular keys may be made by or'ing  */
/* the ntsc or pal MONITOR_ID with the desired MODE_KEY... */

#define LORES_KEY                       0x00000000
#define HIRES_KEY                       0x00008000
#define SUPER_KEY                       0x00008020
#define HAM_KEY                 0x00000800
#define LORESLACE_KEY                   0x00000004
#define HIRESLACE_KEY                   0x00008004
#define SUPERLACE_KEY                   0x00008024
#define HAMLACE_KEY                     0x00000804
#define LORESDPF_KEY                    0x00000400
#define HIRESDPF_KEY                    0x00008400
#define SUPERDPF_KEY                    0x00008420
#define LORESLACEDPF_KEY                0x00000404
#define HIRESLACEDPF_KEY                0x00008404
#define SUPERLACEDPF_KEY                0x00008424
#define LORESDPF2_KEY                   0x00000440
#define HIRESDPF2_KEY                   0x00008440
#define SUPERDPF2_KEY                   0x00008460
#define LORESLACEDPF2_KEY               0x00000444
#define HIRESLACEDPF2_KEY               0x00008444
#define SUPERLACEDPF2_KEY               0x00008464
#define EXTRAHALFBRITE_KEY              0x00000080
#define EXTRAHALFBRITELACE_KEY          0x00000084

/* vga identifiers */

#define VGA_MONITOR_ID                  0x00031000

#define VGAEXTRALORES_KEY               0x00031004
#define VGALORES_KEY                    0x00039004
#define VGAPRODUCT_KEY                  0x00039024
#define VGAHAM_KEY                      0x00031804
#define VGAEXTRALORESLACE_KEY           0x00031005
#define VGALORESLACE_KEY                0x00039005
#define VGAPRODUCTLACE_KEY              0x00039025
#define VGAHAMLACE_KEY                  0x00031805
#define VGAEXTRALORESDPF_KEY            0x00031404
#define VGALORESDPF_KEY         0x00039404
#define VGAPRODUCTDPF_KEY               0x00039424
#define VGAEXTRALORESLACEDPF_KEY        0x00031405
#define VGALORESLACEDPF_KEY             0x00039405
```

```
#define VGAPRODUCTLACEDPF_KEY          0x00039425
#define VGAEXTRALORESDPF2_KEY          0x00031444
#define VGALORESDPF2_KEY               0x00039444
#define VGAPRODUCTDPF2_KEY             0x00039464
#define VGAEXTRALORESLACEDPF2_KEY      0x00031445
#define VGALORESLACEDPF2_KEY           0x00039445
#define VGAPRODUCTLACEDPF2_KEY         0x00039465
#define VGAEXTRAHALFBRITE_KEY          0x00031084
#define VGAEXTRAHALFBRITELACE_KEY      0x00031085

/* a2024 identifiers */

#define A2024_MONITOR_ID               0x00041000

#define A2024TENHERTZ_KEY              0x00041000
#define A2024FIFTEENHERTZ_KEY          0x00049000

/* prototype identifiers */

#define PROTO_MONITOR_ID               0x00051000

#endif  /* GRAPHICS_DISPLAYINFO_H */
```

```
#ifndef GRAPHICS_GELS_H
#define GRAPHICS_GELS_H
/*
**      $Filename: graphics/gels.h $
**      $Release: 2.04 $
**      $Revision: 37.0 $
**      $Date: 91/01/07 $
**
**      include file for AMIGA GELS (Graphics Elements)
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_TYPES_H
#include <exec/types.h>
#endif

/* VSprite flags */
/* user-set VSprite flags: */
#define SUSERFLAGS  0x00FF    /* mask of all user-settable VSprite-flags */
#define VSPRITE     0x0001    /* set if VSprite, clear if Bob */
#define SAVEBACK    0x0002    /* set if background is to be saved/restored */
#define OVERLAY     0x0004    /* set to mask image of Bob onto background */
#define MUSTDRAW    0x0008    /* set if VSprite absolutely must be drawn */
/* system-set VSprite flags: */
#define BACKSAVED   0x0100    /* this Bob's background has been saved */
#define BOBUPDATE   0x0200    /* temporary flag, useless to outside world */
#define GELGONE     0x0400    /* set if gel is completely clipped (offscreen) */
#define VSOVERFLOW  0x0800    /* VSprite overflow (if MUSTDRAW set we draw!) */

/* Bob flags */
/* these are the user flag bits */
#define BUSERFLAGS  0x00FF    /* mask of all user-settable Bob-flags */
#define SAVEBOB     0x0001    /* set to not erase Bob */
#define BOBISCOMP   0x0002    /* set to identify Bob as AnimComp */
/* these are the system flag bits */
#define BWAITING    0x0100    /* set while Bob is waiting on 'after' */
#define BDRAWN      0x0200    /* set when Bob is drawn this DrawG pass*/
#define BOBSAWAY    0x0400    /* set to initiate removal of Bob */
#define BOBNIX      0x0800    /* set when Bob is completely removed */
#define SAVEPRESERVE 0x1000   /* for back-restore during double-buffer*/
#define OUTSTEP     0x2000    /* for double-clearing if double-buffer */

/* defines for the animation procedures */
#define ANFRACSIZE  6
#define ANIMHALF    0x0020
#define RINGTRIGGER 0x0001

/* UserStuff definitions
 *  the user can define these to be a single variable or a sub-structure
 *  if undefined by the user, the system turns these into innocuous variables
 *  see the manual for a thorough definition of the UserStuff definitions
 */
#ifndef VUserStuff            /* VSprite user stuff */
#define VUserStuff WORD
#endif

#ifndef BUserStuff            /* Bob user stuff */
#define BUserStuff WORD
#endif

#ifndef AUserStuff            /* AnimOb user stuff */
#define AUserStuff WORD
#endif

```

```

/*********************** GEL STRUCTURES ***********************************/

struct VSprite
{
/* -------------------- SYSTEM VARIABLES ------------------------------- */
/* GEL linked list forward/backward pointers sorted by y,x value */
    struct VSprite   *NextVSprite;
    struct VSprite   *PrevVSprite;

/* GEL draw list constructed in the order the Bobs are actually drawn, then
 *  list is copied to clear list
 *  must be here in VSprite for system boundary detection
 */
    struct VSprite   *DrawPath;     /* pointer of overlay drawing */
    struct VSprite   *ClearPath;    /* pointer for overlay clearing */

/* the VSprite positions are defined in (y,x) order to make sorting
 *  sorting easier, since (y,x) as a long integer
 */
    WORD OldY, OldX;            /* previous position */

/* -------------------- COMMON VARIABLES -------------------------------- */
    WORD Flags;         /* VSprite flags */

/* -------------------- USER VARIABLES ---------------------------------- */
/* the VSprite positions are defined in (y,x) order to make sorting
 *  sorting easier, since (y,x) as a long integer
 */
    WORD Y, X;                  /* screen position */

    WORD Height;
    WORD Width;         /* number of words per row of image data */
    WORD Depth;         /* number of planes of data */

    WORD MeMask;                /* which types can collide with this VSprite*/
    WORD HitMask;               /* which types this VSprite can collide with*/

    WORD *ImageData;            /* pointer to VSprite image */

/* borderLine is the one-dimensional logical OR of all
 *  the VSprite bits, used for fast collision detection of edge
 */
    WORD *BorderLine;           /* logical OR of all VSprite bits */
    WORD *CollMask;             /* similar to above except this is a matrix */

/* pointer to this VSprite's color definitions (not used by Bobs) */
    WORD *SprColors;

    struct Bob *VSBob;          /* points home if this VSprite is part of
                                   a Bob */

/* planePick flag:  set bit selects a plane from image, clear bit selects
 *  use of shadow mask for that plane
 * OnOff flag: if using shadow mask to fill plane, this bit (corresponding
 *  to bit in planePick) describes whether to fill with 0's or 1's
 * There are two uses for these flags:
 *      - if this is the VSprite of a Bob, these flags describe how the Bob
 *        is to be drawn into memory
 *      - if this is a simple VSprite and the user intends on setting the
 *        MUSTDRAW flag of the VSprite, these flags must be set too to describe
 *        which color registers the user wants for the image
 */
    BYTE PlanePick;
    BYTE PlaneOnOff;

    VUserStuff VUserExt;        /* user definable:  see note above */
};

struct Bob
/* blitter-objects */
{
/* --------------------- SYSTEM VARIABLES --------------------------------- */

/* --------------------- COMMON VARIABLES --------------------------------- */
    WORD Flags; /* general purpose flags (see definitions below) */

/* --------------------- USER VARIABLES ----------------------------------- */
    WORD *SaveBuffer;   /* pointer to the buffer for background save */

/* used by Bobs for "cookie-cutting" and multi-plane masking */
    WORD *ImageShadow;

/* pointer to BOBs for sequenced drawing of Bobs
 *  for correct overlaying of multiple component animations
 */
    struct Bob *Before; /* draw this Bob before Bob pointed to by before */
    struct Bob *After;  /* draw this Bob after Bob pointed to by after */

    struct VSprite   *BobVSprite;   /* this Bob's VSprite definition */

    struct AnimComp  *BobComp;      /* pointer to this Bob's AnimComp def */

    struct DBufPacket *DBuffer;     /* pointer to this Bob's dBuf packet */

    BUserStuff BUserExt;            /* Bob user extension */
};

struct AnimComp
{
/* --------------------- SYSTEM VARIABLES --------------------------------- */

/* --------------------- COMMON VARIABLES --------------------------------- */
    WORD Flags;             /* AnimComp flags for system & user */

/* timer defines how long to keep this component active:
 *  if set non-zero, timer decrements to zero then switches to nextSeq
 *  if set to zero, AnimComp never switches
 */
    WORD Timer;

/* --------------------- USER VARIABLES ----------------------------------- */
/* initial value for timer when the AnimComp is activated by the system */
    WORD TimeSet;

/* pointer to next and previous components of animation object */
    struct AnimComp  *NextComp;
    struct AnimComp  *PrevComp;

/* pointer to component component definition of next image in sequence */
    struct AnimComp  *NextSeq;
    struct AnimComp  *PrevSeq;

    WORD (*AnimCRoutine)(); /* address of special animation procedure */

    WORD YTrans;     /* initial y translation (if this is a component) */
    WORD XTrans;     /* initial x translation (if this is a component) */

    struct AnimOb    *HeadOb;

    struct Bob       *AnimBob;
};
```

```

struct AnimOb
{
/* --------------------- SYSTEM VARIABLES --------------------------------- */
    struct AnimOb    *NextOb, *PrevOb;

/* number of calls to Animate this AnimOb has endured */
    LONG Clock;

    WORD AnOldY, AnOldX;            /* old y,x coordinates */

/* --------------------- COMMON VARIABLES --------------------------------- */
    WORD AnY, AnX;                  /* y,x coordinates of the AnimOb */

/* --------------------- USER VARIABLES ----------------------------------- */
    WORD YVel, XVel;                /* velocities of this object */
    WORD YAccel, XAccel;            /* accelerations of this object */

    WORD RingYTrans, RingXTrans;    /* ring translation values */

    WORD (*AnimORoutine)();         /* address of special animation
                                       procedure */

    struct AnimComp  *HeadComp;     /* pointer to first component */

    AUserStuff AUserExt;            /* AnimOb user extension */
};

/* dBufPacket defines the values needed to be saved across buffer to buffer
 *  when in double-buffer mode
 */
struct DBufPacket
{
    WORD BufY, BufX;                /* save the other buffers screen coordinates
   */
    struct VSprite   *BufPath;      /* carry the draw path over the gap */

/* these pointers must be filled in by the user */
/* pointer to other buffer's background save buffer */
    WORD *BufBuffer;
};


/* ************************************************************************ */

/* these are GEL functions that are currently simple enough to exist as a
 *  definition.  It should not be assumed that this will always be the case
 */
#define InitAnimate(animKey) {*(animKey) = NULL;}
#define RemBob(b) {(b)->Flags |= BOBSAWAY;}

/* ************************************************************************ */

#define B2NORM      0
#define B2SWAP      1
#define B2BOBBER    2

/* ************************************************************************ */

/* a structure to contain the 16 collision procedure addresses */
struct collTable
{
    int (*collPtrs[16])();
};

#endif  /* GRAPHICS_GELS_H */
```

graphics/gfx.h

```
#ifndef GRAPHICS_GFX_H
#define GRAPHICS_GFX_H
/*
**      $Filename: graphics/gfx.h $
**      $Release: 2.04 $
**      $Revision: 37.0 $
**      $Date: 91/01/07 $
**
**      general include file for application programs
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_TYPES_H
#include <exec/types.h>
#endif

#define BITSET  0x8000
#define BITCLR  0

#define AGNUS
#ifdef AGNUS
#define TOBB(a)      ((long)(a))
#else
#define TOBB(a)      ((long)(a)>>1)  /* convert Chip adr to Bread Board Adr */
#endif

struct Rectangle
{
    WORD   MinX,MinY;
    WORD   MaxX,MaxY;
};

struct Rect32
{
    LONG    MinX,MinY;
    LONG    MaxX,MaxY;
};

typedef struct tPoint
{
    WORD x,y;
} Point;

typedef UBYTE *PLANEPTR;

struct BitMap
{
    UWORD   BytesPerRow;
    UWORD   Rows;
    UBYTE   Flags;
    UBYTE   Depth;
    UWORD   pad;
    PLANEPTR Planes[8];
};

#define RASSIZE(w,h)    ((h)*( ((w)+15)>>3&0xFFFE))

#endif  /* GRAPHICS_GFX_H */
```

graphics/gfxbase.h

```
#ifndef GRAPHICS_GFXBASE_H
#define GRAPHICS_GFXBASE_H
/*
**      $Filename: graphics/gfxbase.h $
**      $Release: 2.04 $
**      $Revision: 37.2 $
**      $Date: 91/02/07 $
**
**      graphics base definitions
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_LISTS_H
#include <exec/lists.h>
#endif
#ifndef EXEC_LIBRARIES_H
#include <exec/libraries.h>
#endif
#ifndef EXEC_INTERRUPTS_H
#include <exec/interrupts.h>
#endif

struct GfxBase
{
    struct Library  LibNode;
    struct View *ActiView;
    struct copinit *copinit;    /* ptr to copper start up list */
    long    *cia;               /* for 8520 resource use */
    long    *blitter;           /* for future blitter resource use */
    UWORD   *LOFlist;
    UWORD   *SHFlist;
    struct bltnode *blthd,*bltt1;
    struct bltnode *bsblthd,*bsbltt1;
    struct Interrupt vbsrv,timsrv,bltsrv;
    struct List     TextFonts;
    struct  TextFont *DefaultFont;
    UWORD  Modes;                       /* copy of current first bplcon0 */
    BYTE VBlank;
    BYTE Debug;
    WORD BeamSync;
    WORD system_bplcon0;        /* it is ored into each bplcon0 for display */
    UBYTE   SpriteReserved;
    UBYTE   bytereserved;
    UWORD   Flags;
    WORD   BlitLock;
        WORD    BlitNest;

        struct  List    BlitWaitQ;
        struct  Task    *BlitOwner;
        struct  List    TOF_WaitQ;
        UWORD   DisplayFlags;           /* NTSC PAL GENLOC etc*/
                                        /* flags initialized at power on */
        struct SimpleSprite **SimpleSprites;
        UWORD   MaxDisplayRow;          /* hardware stuff, do not use */
        UWORD   MaxDisplayColumn;       /* hardware stuff, do not use */
        UWORD   NormalDisplayRows;
        UWORD   NormalDisplayColumns;
        /* the following are for standard non interlace, 1/2 wb width */
        UWORD   NormalDPMX;             /* Dots per meter on display */
        UWORD   NormalDPMY;             /* Dots per meter on display */
        struct  SignalSemaphore *LastChanceMemory;
        UWORD   *LCMptr;
        UWORD   MicrosPerLine;          /* 256 time usec/line */
        UWORD   MinDisplayColumn;
```

graphics/gfxbase.h

```
        UBYTE   ChipRevBits0;
        UBYTE   crb_reserved[5];
        UWORD   monitor_id;             /* normally null */
        ULONG   hedley[8];
        ULONG   hedley_sprites[8] ;     /* sprite ptrs for intuition mouse */
        ULONG   hedley_sprites1[8] ;            /* sprite ptrs for intuition mou
    se */
        WORD    hedley_count;
        UWORD   hedley_flags;
        WORD    hedley_tmp;
        LONG    *hash_table;
        UWORD   current_tot_rows;
        UWORD   current_tot_cclks;
        UBYTE   hedley_hint;
        UBYTE   hedley_hint2;
        ULONG   nreserved[4];
        LONG    *a2024_sync_raster;
        WORD   control_delta_pal;
        WORD   control_delta_ntsc;
        struct  MonitorSpec *current_monitor;
        struct  List MonitorList;
        struct  MonitorSpec *default_monitor;
        struct  SignalSemaphore *MonitorListSemaphore;
        VOID    *DisplayInfoDataBase;
        struct  SignalSemaphore *ActiViewCprSemaphore;
        ULONG   *UtilityBase;           /* for hook and tag utilities   */
        ULONG   *ExecBase;              /* to link with rom.lib */
        };

#define NTSC            1
#define GENLOC          2
#define PAL             4
#define TODA_SAFE       8

#define BLITMSG_FAULT   4

/* bits defs for ChipRevBits */
#define GFXB_BIG_BLITS  0
#define GFXB_HR_AGNUS   0
#define GFXB_HR_DENISE  1

#define GFXF_BIG_BLITS  1
#define GFXF_HR_AGNUS   1
#define GFXF_HR_DENISE  2

#define GRAPHICSNAME    "graphics.library"

#endif  /* GRAPHICS_GFXBASE_H */
```

graphics/gfxmacros.h

```
#ifndef GRAPHICS_GFXMACROS_H
#define GRAPHICS_GFXMACROS_H
/*
**      $Filename: graphics/gfxmacros.h $
**      $Release: 2.04 $
**      $Revision: 37.0 $
**      $Date: 91/01/07 $
**
**
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_TYPES_H
#include <exec/types.h>
#endif

#ifndef  GRAPHICS_RASTPORT_H
#include <graphics/rastport.h>
#endif

#define ON_DISPLAY      custom.dmacon = BITSET|DMAF_RASTER;
#define OFF_DISPLAY     custom.dmacon = BITCLR|DMAF_RASTER;
#define ON_SPRITE       custom.dmacon = BITSET|DMAF_SPRITE;
#define OFF_SPRITE      custom.dmacon = BITCLR|DMAF_SPRITE;

#define ON_VBLANK       custom.intena = BITSET|INTF_VERTB;
#define OFF_VBLANK      custom.intena = BITCLR|INTF_VERTB;

#define SetOPen(w,c)    {(w)->AOlPen = c;(w)->Flags |= AREAOUTLINE;}
#define SetDrPt(w,p)    {(w)->LinePtrn = p;(w)->Flags |= FRST_DOT;(w)->linpatcnt
    =15;}
#define SetWrMsk(w,m)   {(w)->Mask = m;}
#define SetAfPt(w,p,n)  {(w)->AreaPtrn = p;(w)->AreaPtSz = n;}

#define BNDRYOFF(w)     {(w)->Flags &= ~AREAOUTLINE;}

#define CINIT(c,n)        UCopperListInit(c,n);
#define CMOVE(c,a,b)    { CMove(c,&a,b);CBump(c); }
#define CWAIT(c,a,b)    { CWait(c,a,b);CBump(c); }
#define CEND(c) { CWAIT(c,10000,255); }

#define DrawCircle(rp,cx,cy,r)  DrawEllipse(rp,cx,cy,r,r);
#define AreaCircle(rp,cx,cy,r)  AreaEllipse(rp,cx,cy,r,r);

#endif  /* GRAPHICS_GFXMACROS_H */
```

graphics/gfxnodes.h

Page 1

```
#ifndef GRAPHICS_GFXNODES_H
#define GRAPHICS_GFXNODES_H
/*
**      $Filename: graphics/gfxnodes.h $
**      $Release: 2.04 $
**      $Revision: 37.0 $
**      $Date: 91/01/07 $
**
**      graphics extended node definintions
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_NODES_H
#include <exec/nodes.h>
#endif

struct  ExtendedNode    {
struct  Node    *xln_Succ;
struct  Node    *xln_Pred;
UBYTE   xln_Type;
BYTE    xln_Pri;
char    *xln_Name;
UBYTE   xln_Subsystem;
UBYTE   xln_Subtype;
LONG    xln_Library;
LONG    (*xln_Init)();
};

#define SS_GRAPHICS     0x02

#define VIEW_EXTRA_TYPE         1
#define VIEWPORT_EXTRA_TYPE     2
#define SPECIAL_MONITOR_TYPE    3
#define MONITOR_SPEC_TYPE       4

#endif  /* GRAPHICS_GFXNODES_H */
```

graphics/graphint.h

Page 1

```
#ifndef GRAPHICS_GRAPHINT_H
#define GRAPHICS_GRAPHINT_H
/*
**      $Filename: graphics/graphint.h $
**      $Release: 2.04 $
**      $Revision: 37.0 $
**      $Date: 91/01/07 $
**
**
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_NODES_H
#include <exec/nodes.h>
#endif

/* structure used by AddTOFTask */
struct Isrvstr
{
    struct Node is_Node;
    struct Isrvstr *Iptr;   /* passed to srvr by os */
    int (*code)();
    int (*ccode)();
    int Carg;
};

#endif  /* GRAPHICS_GRAPHINT_H */
```

graphics/layers.h

```
#ifndef GRAPHICS_LAYERS_H
#define GRAPHICS_LAYERS_H
/*
**      $Filename: graphics/layers.h $
**      $Release: 2.04 $
**      $Revision: 37.0 $
**      $Date: 91/01/07 $
**
**
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_LISTS_H
#include <exec/lists.h>
#endif

#ifndef EXEC_SEMAPHORES_H
#include <exec/semaphores.h>
#endif

#define LAYERSIMPLE     1
#define LAYERSMART      2
#define LAYERSUPER      4
#define LAYERUPDATING   0x10
#define LAYERBACKDROP   0x40
#define LAYERREFRESH    0x80
#define LAYER_CLIPRECTS_LOST    0x100   /* during BeginUpdate */
                                        /* or during layerop */
                                        /* this happens if out of memory */
#define LMN_REGION      -1

struct Layer_Info
{
    struct Layer *top_layer;
    struct Layer *check_lp;         /* system use */
    struct ClipRect *obs;           /* system use */
        struct MinList  FreeClipRects;
        struct SignalSemaphore Lock;
        struct List gs_Head;                    /* system use */
        LONG    longreserved;
        UWORD   Flags;
        BYTE    fatten_count;
        BYTE    LockLayersCount;
    UWORD  LayerInfo_extra_size;
    WORD        *blitbuff;
    VOID        *LayerInfo_extra;
};

#define NEWLAYERINFO_CALLED 1
#define ALERTLAYERSNOMEM 0x83010000

#endif  /* GRAPHICS_LAYERS_H */
```

---

graphics/monitor.h

```
#ifndef GRAPHICS_MONITOR_H
#define GRAPHICS_MONITOR_H
/*
**      $Filename: graphics/monitor.h $
**      $Release: 2.04 $
**      $Revision: 37.0 $
**      $Date: 91/01/07 $
**
**      graphics monitorspec definintions
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_SEMAPHORES_H
#include        <exec/semaphores.h>
#endif

#ifndef GRAPHICS_GFXNODES_H
#include        <graphics/gfxnodes.h>
#endif

#ifndef GRAPHICS_GFX_H
#include        <graphics/gfx.h>
#endif

struct  MonitorSpec
{
    struct      ExtendedNode    ms_Node;
    UWORD       ms_Flags;
    LONG        ratioh;
    LONG        ratiov;
    UWORD       total_rows;
    UWORD       total_colorclocks;
    UWORD       DeniseMaxDisplayColumn;
    UWORD       BeamCon0;
    UWORD       min_row;
    struct      SpecialMonitor  *ms_Special;
    UWORD       ms_OpenCount;
    LONG        (*ms_transform)();
    LONG        (*ms_translate)();
    LONG        (*ms_scale)();
    UWORD       ms_xoffset;
    UWORD       ms_yoffset;
    struct      Rectangle       ms_LegalView;
    LONG        (*ms_maxoscan)();       /* maximum legal overscan */
    LONG        (*ms_videoscan)();      /* video display overscan */
    UWORD       DeniseMinDisplayColumn;
    ULONG       DisplayCompatible;
    struct      List DisplayInfoDataBase;
    struct      SignalSemaphore DisplayInfoDataBaseSemaphore;
    ULONG       ms_reserved00;
    ULONG       ms_reserved01;
};

#define TO_MONITOR      0
#define FROM_MONITOR    1
#define STANDARD_XOFFSET        9
#define STANDARD_YOFFSET        0
#define REQUEST_NTSC    1
#define REQUEST_PAL     2
#define REQUEST_SPECIAL 4
#define REQUEST_A2024   8

#define DEFAULT_MONITOR_NAME    "default.monitor"
#define NTSC_MONITOR_NAME       "ntsc.monitor"
```

```
#define PAL_MONITOR_NAME        "pal.monitor"
#define STANDARD_MONITOR_MASK   ( REQUEST_NTSC | REQUEST_PAL )

#define STANDARD_NTSC_ROWS      262
#define STANDARD_PAL_ROWS       312
#define STANDARD_COLORCLOCKS    226
#define STANDARD_DENISE_MAX     455
#define STANDARD_DENISE_MIN     93
#define STANDARD_NTSC_BEAMCON   ( 0x0000 )
#define STANDARD_PAL_BEAMCON    ( DISPLAYPAL )

#define SPECIAL_BEAMCON ( VARVBLANK | LOLDIS | VARVSYNC | VARBEAM | CSBLANK )

#define MIN_NTSC_ROW    21
#define MIN_PAL_ROW     29
#define STANDARD_VIEW_X 0x81
#define STANDARD_VIEW_Y 0x2C
#define STANDARD_HBSTRT 0x06
#define STANDARD_HSSTRT 0x0B
#define STANDARD_HSSTOP 0x1C
#define STANDARD_HBSTOP 0x2C
#define STANDARD_VBSTRT 0x0122
#define STANDARD_VSSTRT 0x02A6
#define STANDARD_VSSTOP 0x03AA
#define STANDARD_VBSTOP 0x1066

#define VGA_COLORCLOCKS (STANDARD_COLORCLOCKS/2)
#define VGA_TOTAL_ROWS  (STANDARD_NTSC_ROWS*2)
#define VGA_DENISE_MIN  59
#define MIN_VGA_ROW     29
#define VGA_HBSTRT      0x08
#define VGA_HSSTRT      0x0E
#define VGA_HSSTOP      0x1C
#define VGA_HBSTOP      0x1E
#define VGA_VBSTRT      0x0000
#define VGA_VSSTRT      0x0153
#define VGA_VSSTOP      0x0235
#define VGA_VBSTOP      0x0CCD

#define VGA_MONITOR_NAME        "vga.monitor"

#define VGA70_COLORCLOCKS (STANDARD_COLORCLOCKS/2)
#define VGA70_TOTAL_ROWS 449
#define VGA70_DENISE_MIN 59
#define MIN_VGA70_ROW    35
#define VGA70_HBSTRT     0x08
#define VGA70_HSSTRT     0x0E
#define VGA70_HSSTOP     0x1C
#define VGA70_HBSTOP     0x1E
#define VGA70_VBSTRT     0x0000
#define VGA70_VSSTRT     0x02A6
#define VGA70_VSSTOP     0x0388
#define VGA70_VBSTOP     0x0F73

#define VGA70_BEAMCON    (SPECIAL_BEAMCON ^ VSYNCTRUE)
#define VGA70_MONITOR_NAME      "vga70.monitor"

#define BROADCAST_HBSTRT        0x01
#define BROADCAST_HSSTRT        0x06
#define BROADCAST_HSSTOP        0x17
#define BROADCAST_HBSTOP        0x27
#define BROADCAST_VBSTRT        0x0000
#define BROADCAST_VSSTRT        0x02A6
#define BROADCAST_VSSTOP        0x054C
#define BROADCAST_VBSTOP        0x1C40
#define BROADCAST_BEAMCON       ( LOLDIS | CSBLANK )
```

```
#define RATIO_FIXEDPART 4
#define RATIO_UNITY     (1 << RATIO_FIXEDPART)

struct  AnalogSignalInterval
{
    UWORD       asi_Start;
    UWORD       asi_Stop;
};

struct  SpecialMonitor
{
    struct      ExtendedNode    spm_Node;
    UWORD       spm_Flags;
    int (*do_monitor)();
    int (*reserved1)();
    int (*reserved2)();
    int (*reserved3)();
    struct      AnalogSignalInterval    hblank;
    struct      AnalogSignalInterval    vblank;
    struct      AnalogSignalInterval    hsync;
    struct      AnalogSignalInterval    vsync;
};

#endif  /* GRAPHICS_MONITOR_H */
```

graphics/rastport.h

```
#ifndef GRAPHICS_RASTPORT_H
#define GRAPHICS_RASTPORT_H
/*
**      $Filename: graphics/rastport.h $
**      $Release: 2.04 $
**      $Revision: 37.0 $
**      $Date: 91/01/07 $
**
**
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_TYPES_H
#include <exec/types.h>
#endif

#ifndef GRAPHICS_GFX_H
#include <graphics/gfx.h>
#endif

struct AreaInfo
{
    WORD   *VctrTbl;          /* ptr to start of vector table */
    WORD   *VctrPtr;          /* ptr to current vertex */
    BYTE    *FlagTbl;          /* ptr to start of vector flag table */
    BYTE    *FlagPtr;          /* ptrs to areafill flags */
    WORD   Count;             /* number of vertices in list */
    WORD   MaxCount;          /* AreaMove/Draw will not allow Count>MaxCount*/
    WORD   FirstX,FirstY;     /* first point for this polygon */
};

struct TmpRas
{
    BYTE *RasPtr;
    LONG Size;
};

/* unoptimized for 32bit alignment of pointers */
struct GelsInfo
{
    BYTE sprRsrvd;             /* flag of which sprites to reserve from
                                  vsprite system */
    UBYTE Flags;               /* system use */
    struct VSprite *gelHead, *gelTail; /* dummy vSprites for list management*/
    /* pointer to array of 8 WORDS for sprite available lines */
    WORD *nextLine;
    /* pointer to array of 8 pointers for color-last-assigned to vSprites */
    WORD **lastColor;
    struct collTable *collHandler;     /* addresses of collision routines */
    WORD leftmost, rightmost, topmost, bottommost;
   APTR firstBlissObj,lastBlissObj;    /* system use only */
};

struct RastPort
{
    struct  Layer *Layer;
    struct  BitMap   *BitMap;
    UWORD  *AreaPtrn;           /* ptr to areafill pattern */
    struct  TmpRas *TmpRas;
    struct  AreaInfo *AreaInfo;
    struct  GelsInfo *GelsInfo;
    UBYTE   Mask;              /* write mask for this raster */
    BYTE    FgPen;             /* foreground pen for this raster */
    BYTE    BgPen;             /* background pen  */
```

```
    BYTE    AOlPen;            /* areafill outline pen */
    BYTE    DrawMode;          /* drawing mode for fill, lines, and text */
    BYTE    AreaPtSz;          /* 2^n words for areafill pattern */
    BYTE    linpatcnt;         /* current line drawing pattern preshift */
    BYTE    dummy;
    UWORD  Flags;             /* miscellaneous control bits */
    UWORD  LinePtrn;          /* 16 bits for textured lines */
    WORD   cp_x, cp_y;        /* current pen position */
    UBYTE   minterms[8];
    WORD   PenWidth;
    WORD   PenHeight;
    struct  TextFont *Font;    /* current font address */
    UBYTE   AlgoStyle;         /* the algorithmically generated style */
    UBYTE   TxFlags;           /* text specific flags */
    UWORD   TxHeight;          /* text height */
    UWORD   TxWidth;           /* text nominal width */
    UWORD   TxBaseline;        /* text baseline */
    WORD    TxSpacing;         /* text spacing (per character) */
    APTR    *RP_User;
    ULONG   longreserved[2];
#ifndef GFX_RASTPORT_1_2
    UWORD   wordreserved[7];   /* used to be a node */
    UBYTE   reserved[8];       /* for future use */
#endif
};

/* drawing modes */
#define JAM1        0          /* jam 1 color into raster */
#define JAM2        1          /* jam 2 colors into raster */
#define COMPLEMENT  2          /* XOR bits into raster */
#define INVERSVID   4          /* inverse video for drawing modes */

/* these are the flag bits for RastPort flags */
#define FRST_DOT    0x01       /* draw the first dot of this line ? */
#define ONE_DOT     0x02       /* use one dot mode for drawing lines */
#define DBUFFER     0x04       /* flag set when RastPorts
                                  are double-buffered */

           /* only used for bobs */

#define AREAOUTLINE 0x08       /* used by areafiller */
#define NOCROSSFILL 0x20       /* areafills have no crossovers */

/* there is only one style of clipping: raster clipping */
/* this preserves the continuity of jaggies regardless of clip window */
/* When drawing into a RastPort, if the ptr to ClipRect is nil then there */
/* is no clipping done, this is dangerous but useful for speed */

#endif  /* GRAPHICS_RASTPORT_H */
```

graphics/regions.h

```
#ifndef GRAPHICS_REGIONS_H
#define GRAPHICS_REGIONS_H
/*
**      $Filename: graphics/regions.h $
**      $Release: 2.04 $
**      $Revision: 37.0 $
**      $Date: 91/01/07 $
**
**
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_TYPES_H
#include <exec/types.h>
#endif

#ifndef GRAPHICS_GFX_H
#include <graphics/gfx.h>
#endif

struct RegionRectangle
{
    struct RegionRectangle *Next,*Prev;
    struct Rectangle bounds;
};

struct Region
{
    struct Rectangle bounds;
    struct RegionRectangle *RegionRectangle;
};

#endif  /* GRAPHICS_REGIONS_H */
```

graphics/scale.h

```
#ifndef GRAPHICS_SCALE_H
#define GRAPHICS_SCALE_H
/*
**      $Filename: graphics/scale.h $
**      $Release: 2.04 $
**      $Revision: 37.0 $
**      $Date: 91/01/07 $
**
**      structure argument to BitMapScale()
**
**      (C) Copyright 1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_TYPES_H
#include <exec/types.h>
#endif

struct BitScaleArgs {
    UWORD   bsa_SrcX, bsa_SrcY;                 /* source origin */
    UWORD   bsa_SrcWidth, bsa_SrcHeight;        /* source size */
    UWORD   bsa_XSrcFactor, bsa_YSrcFactor;     /* scale factor denominators */
    UWORD   bsa_DestX, bsa_DestY;               /* destination origin */
    UWORD   bsa_DestWidth, bsa_DestHeight;      /* destination size result */
    UWORD   bsa_XDestFactor, bsa_YDestFactor;   /* scale factor numerators */
    struct BitMap *bsa_SrcBitMap;               /* source BitMap */
    struct BitMap *bsa_DestBitMap;              /* destination BitMap */
    ULONG   bsa_Flags;                          /* reserved.  Must be zero! */
    UWORD   bsa_XDDA, bsa_YDDA;                 /* reserved */
    LONG    bsa_Reserved1;
    LONG    bsa_Reserved2;
};
#endif  /* GRAPHICS_SCALE_H */
```

graphics/sprite.h

```
#ifndef GRAPHICS_SPRITE_H
#define GRAPHICS_SPRITE_H
/*
**      $Filename: graphics/sprite.h $
**      $Release: 2.04 $
**      $Revision: 37.0 $
**      $Date: 91/01/07 $
**
**
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_TYPES_H
#include <exec/types.h>
#endif

#define SPRITE_ATTACHED 0x80

struct SimpleSprite
{
    UWORD *posctldata;
    UWORD height;
    UWORD   x,y;    /* current position */
    UWORD   num;
};

#endif  /* GRAPHICS_SPRITE_H */
```

graphics/text.h

```
#ifndef GRAPHICS_TEXT_H
#define GRAPHICS_TEXT_H
/*
**      $Filename: graphics/text.h $
**      $Release: 2.04 $
**      $Revision: 37.0 $
**      $Date: 91/01/07 $
**
**      graphics library text structures
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_PORTS_H
#include        "exec/ports.h"
#endif  /* EXEC_PORTS_H */

#ifndef GRAPHICS_GFX_H
#include        "graphics/gfx.h"
#endif  /* GRAPHICS_GFX_H */

#ifndef UTILITY_TAGITEM_H
#include        "utility/tagitem.h"
#endif  /* UTILITY_TAGITEM_H */

/*------ Font Styles ------------------------------------------------*/
#define FS_NORMAL        0       /* normal text (no style bits set) */
#define FSB_UNDERLINED   0       /* underlined (under baseline) */
#define FSF_UNDERLINED   0x01
#define FSB_BOLD         1       /* bold face text (ORed w/ shifted) */
#define FSF_BOLD         0x02
#define FSB_ITALIC       2       /* italic (slanted 1:2 right) */
#define FSF_ITALIC       0x04
#define FSB_EXTENDED     3       /* extended face (wider than normal) */
#define FSF_EXTENDED     0x08

#define FSB_COLORFONT    6       /* this uses ColorTextFont structure */
#define FSF_COLORFONT    0x40
#define FSB_TAGGED       7       /* the TextAttr is really an TTextAttr, */
#define FSF_TAGGED       0x80

/*------ Font Flags -------------------------------------------------*/
#define FPB_ROMFONT      0       /* font is in rom */
#define FPF_ROMFONT      0x01
#define FPB_DISKFONT     1       /* font is from diskfont.library */
#define FPF_DISKFONT     0x02
#define FPB_REVPATH      2       /* designed path is reversed (e.g. left) */
#define FPF_REVPATH      0x04
#define FPB_TALLDOT      3       /* designed for hires non-interlaced */
#define FPF_TALLDOT      0x08
#define FPB_WIDEDOT      4       /* designed for lores interlaced */
#define FPF_WIDEDOT      0x10
#define FPB_PROPORTIONAL 5       /* character sizes can vary from nominal */
#define FPF_PROPORTIONAL 0x20
#define FPB_DESIGNED     6       /* size explicitly designed, not constructed */
                                 /* note: if you do not set this bit in your */
                                 /* textattr, then a font may be constructed */
                                 /* for you by scaling an existing rom or disk */
                                 /* font (under V36 and above). */
#define FPF_DESIGNED     0x40
    /* bit 7 is always clear for fonts on the graphics font list */
#define FPB_REMOVED      7       /* the font has been removed */
#define FPF_REMOVED      (1<<7)

/****** TextAttr node, matches text attributes in RastPort **********/
```

```
struct TextAttr {
    STRPTR   ta_Name;               /* name of the font */
    UWORD    ta_YSize;              /* height of the font */
    UBYTE    ta_Style;              /* intrinsic font style */
    UBYTE    ta_Flags;              /* font preferences and flags */
};

struct TTextAttr {
    STRPTR   tta_Name;              /* name of the font */
    UWORD    tta_YSize;             /* height of the font */
    UBYTE    tta_Style;             /* intrinsic font style */
    UBYTE    tta_Flags;             /* font preferences and flags */
    struct TagItem *tta_Tags;       /* extended attributes */
};


/****** Text Tags ******************************************************/
#define TA_DeviceDPI        (1|TAG_USER)      /* Tag value is Point union: */
                                              /* Hi word XDPI, Lo word YDPI */

#define MAXFONTMATCHWEIGHT          32767     /* perfect match from WeighTAMatch */


/****** TextFonts node *************************************************/
struct TextFont {
    struct Message tf_Message;      /* reply message for font removal */
                                    /* font name in LN            \     used in this */
    UWORD    tf_YSize;              /* font height                |     order to best */
    UBYTE    tf_Style;              /* font style                 |     match a font */
    UBYTE    tf_Flags;              /* preferences and flags      /     request. */
    UWORD    tf_XSize;              /* nominal font width */
    UWORD    tf_Baseline;           /* distance from the top of char to baseline */
    UWORD    tf_BoldSmear;          /* smear to affect a bold enhancement */

    UWORD    tf_Accessors;          /* access count */

    UBYTE    tf_LoChar;             /* the first character described here */
    UBYTE    tf_HiChar;             /* the last character described here */
    APTR     tf_CharData;           /* the bit character data */

    UWORD    tf_Modulo;             /* the row modulo for the strike font data */
    APTR     tf_CharLoc;            /* ptr to location data for the strike font */
                                    /*   2 words: bit offset then size */
    APTR     tf_CharSpace;          /* ptr to words of proportional spacing data */
    APTR     tf_CharKern;           /* ptr to words of kerning data */
};

/* unfortunately, this needs to be explicitly typed */
#define tf_Extension       tf_Message.mn_ReplyPort

/*----- tfe_Flags0 (partial definition) -------------------------------*/
#define TEOB_NOREMFONT     0        /* disallow RemFont for this font */
#define TEOF_NOREMFONT     0x01

struct TextFontExtension {          /* this structure is read-only */
    UWORD    tfe_MatchWord;                  /* a magic cookie for the extension */
    UBYTE    tfe_Flags0;                     /* (system private flags) */
    UBYTE    tfe_Flags1;                     /* (system private flags) */
    struct TextFont *tfe_BackPtr;            /* validation of compilation */
    struct MsgPort *tfe_OrigReplyPort;       /* original value in tf_Extension */
    struct TagItem *tfe_Tags;                /* Text Tags for the font */
    UWORD   *tfe_OFontPatchS;                /* (system private use) */
    UWORD   *tfe_OFontPatchK;                /* (system private use) */
    /* this space is reserved for future expansion */
};
```

```
/****** ColorTextFont node *********************************************/
/*----- ctf_Flags -----------------------------------------------------*/
#define CT_COLORMASK       0x000F  /* mask to get to following color styles */
#define CT_COLORFONT       0x0001  /* color map contains designer's colors */
#define CT_GREYFONT        0x0002  /* color map describes even-stepped */
                                   /* brightnesses from low to high */
#define CT_ANTIALIAS       0x0004  /* zero background thru fully saturated char */

#define CTB_MAPCOLOR       0       /* map ctf_FgColor to the rp_FgPen if it's */
#define CTF_MAPCOLOR       0x0001  /* is a valid color within ctf_Low..ctf_High */

/*----- ColorFontColors -----------------------------------------------*/
struct ColorFontColors {
    UWORD    cfc_Reserved;         /* *must* be zero */
    UWORD    cfc_Count;            /* number of entries in cfc_ColorTable */
    UWORD   *cfc_ColorTable;       /* 4 bit per component color map packed xRGB */
};

/*----- ColorTextFont -------------------------------------------------*/
struct ColorTextFont {
    struct TextFont ctf_TF;
    UWORD    ctf_Flags;            /* extended flags */
    UBYTE    ctf_Depth;            /* number of bit planes */
    UBYTE    ctf_FgColor;          /* color that is remapped to FgPen */
    UBYTE    ctf_Low;              /* lowest color represented here */
    UBYTE    ctf_High;             /* highest color represented here */
    UBYTE    ctf_PlanePick;        /* PlanePick ala Images */
    UBYTE    ctf_PlaneOnOff;       /* PlaneOnOff ala Images */
    struct ColorFontColors *ctf_ColorFontColors; /* colors for font */
    APTR     ctf_CharData[8];      /*pointers to bit planes ala tf_CharData */
};

/****** TextExtent node ************************************************/
struct TextExtent {
    UWORD    te_Width;             /* same as TextLength */
    UWORD    te_Height;            /* same as tf_YSize */
    struct Rectangle te_Extent; /* relative to CP */
};

#endif  /* GRAPHICS_TEXT_H */
```

graphics/videocontrol.h

```
#ifndef GRAPHICS_VIDEOCONTROL_H
#define GRAPHICS_VIDEOCONTROL_H
/*
**      $Filename: graphics/videocontrol.h $
**      $Release: 2.04 $
**      $Revision: 37.0 $
**      $Date: 91/01/07 $
**
**      include define file for videocontrol commands
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_TYPES_H
#include <exec/types.h>
#endif /* EXEC_TYPES_H */

#ifndef UTILITY_TAGITEM_H
#include <utility/tagitem.h>
#endif /* UTILITY_TAGITEM_H */

#define VTAG_END_CM               0x00000000
#define VTAG_CHROMAKEY_CLR        0x80000000
#define VTAG_CHROMAKEY_SET        0x80000001
#define VTAG_BITPLANEKEY_CLR      0x80000002
#define VTAG_BITPLANEKEY_SET      0x80000003
#define VTAG_BORDERBLANK_CLR      0x80000004
#define VTAG_BORDERBLANK_SET      0x80000005
#define VTAG_BORDERNOTRANS_CLR    0x80000006
#define VTAG_BORDERNOTRANS_SET    0x80000007
#define VTAG_CHROMA_PEN_CLR       0x80000008
#define VTAG_CHROMA_PEN_SET       0x80000009
#define VTAG_CHROMA_PLANE_SET     0x8000000A
#define VTAG_ATTACH_CM_SET        0x8000000B
#define VTAG_NEXTBUF_CM           0x8000000C
#define VTAG_BATCH_CM_CLR         0x8000000D
#define VTAG_BATCH_CM_SET         0x8000000E
#define VTAG_NORMAL_DISP_GET      0x8000000F
#define VTAG_NORMAL_DISP_SET      0x80000010
#define VTAG_COERCE_DISP_GET      0x80000011
#define VTAG_COERCE_DISP_SET      0x80000012
#define VTAG_VIEWPORTEXTRA_GET    0x80000013
#define VTAG_VIEWPORTEXTRA_SET    0x80000014
#define VTAG_CHROMAKEY_GET        0x80000015
#define VTAG_BITPLANEKEY_GET      0x80000016
#define VTAG_BORDERBLANK_GET      0x80000017
#define VTAG_BORDERNOTRANS_GET    0x80000018
#define VTAG_CHROMA_PEN_GET       0x80000019
#define VTAG_CHROMA_PLANE_GET     0x8000001A
#define VTAG_ATTACH_CM_GET        0x8000001B
#define VTAG_BATCH_CM_GET         0x8000001C
#define VTAG_BATCH_ITEMS_GET      0x8000001D
#define VTAG_BATCH_ITEMS_SET      0x8000001E
#define VTAG_BATCH_ITEMS_ADD      0x8000001F
#define VTAG_VPMODEID_GET         0x80000020
#define VTAG_VPMODEID_SET         0x80000021
#define VTAG_VPMODEID_CLR         0x80000022
#define VTAG_USERCLIP_GET         0x80000023
#define VTAG_USERCLIP_SET         0x80000024
#define VTAG_USERCLIP_CLR         0x80000025

#endif  /* GRAPHICS_VIDEOCONTROL_H */
```

graphics/view.h

```
#ifndef GRAPHICS_VIEW_H
#define GRAPHICS_VIEW_H
/*
**      $Filename: graphics/view.h $
**      $Release: 2.04 $
**      $Revision: 37.0 $
**      $Date: 91/01/07 $
**
**      graphics view/viewport definintions
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#define ECS_SPECIFIC

#ifndef EXEC_TYPES_H
#include <exec/types.h>
#endif

#ifndef GRAPHICS_GFX_H
#include <graphics/gfx.h>
#endif

#ifndef GRAPHICS_COPPER_H
#include <graphics/copper.h>
#endif

#ifndef GRAPHICS_GFXNODES_H
#include <graphics/gfxnodes.h>
#endif

#ifndef GRAPHICS_MONITOR_H
#include <graphics/monitor.h>
#endif

#ifndef HARDWARE_CUSTOM_H
#include <hardware/custom.h>
#endif

struct ViewPort
{
   struct   ViewPort *Next;
   struct   ColorMap  *ColorMap;    /* table of colors for this viewport */
                 /* if this is nil, MakeVPort assumes default values */
   struct   CopList  *DspIns;       /* user by MakeView() */
   struct   CopList  *SprIns;       /* used by sprite stuff */
   struct   CopList  *ClrIns;       /* used by sprite stuff */
   struct   UCopList *UCopIns;      /* User copper list */
   WORD    DWidth,DHeight;
   WORD    DxOffset,DyOffset;
   UWORD    Modes;
   UBYTE        SpritePriorities;               /* used by makevp */
   UBYTE        ExtendedModes;
   struct   RasInfo *RasInfo;
};

struct View
{
   struct ViewPort *ViewPort;
   struct cprlist *LOFCprList;   /* used for interlaced and noninterlaced */
   struct cprlist *SHFCprList;   /* only used during interlace */
   WORD DyOffset,DxOffset;   /* for complete View positioning */
                              /* offsets are +- adjustments to standard #s */
   UWORD   Modes;             /* such as INTERLACE, GENLOC */
};
```

```

/* these structures are obtained via GfxNew */
/* and disposed by GfxFree */
struct ViewExtra
{
        struct ExtendedNode n;
        struct View *View;       /* backwards link */
        struct MonitorSpec *Monitor; /* monitors for this view */
};

/* this structure is obtained via GfxNew */
/* and disposed by GfxFree */
struct ViewPortExtra
{
        struct ExtendedNode n;
        struct ViewPort *ViewPort;      /* backwards link */
        struct Rectangle DisplayClip;  /* makevp display clipping information */
};

#define EXTEND_VSTRUCT  0x1000  /* unused bit in Modes field of View */

/* defines used for Modes in IVPargs */

#define GENLOCK_VIDEO   0x0002
#define LACE            0x0004
#define SUPERHIRES      0x0020
#define PFBA            0x0040
#define EXTRA_HALFBRITE 0x0080
#define GENLOCK_AUDIO   0x0100
#define DUALPF          0x0400
#define HAM             0x0800
#define EXTENDED_MODE   0x1000
#define VP_HIDE 0x2000
#define SPRITES 0x4000
#define HIRES           0x8000

#define VPF_A2024               0x40
#define VPF_AGNUS               0x20
#define VPF_TENHZ               0x20

struct RasInfo  /* used by callers to and InitDspC() */
{
   struct   RasInfo *Next;          /* used for dualpf */
   struct   BitMap *BitMap;
   WORD    RxOffset,RyOffset;      /* scroll offsets in this BitMap */
};

struct ColorMap
{
        UBYTE   Flags;
        UBYTE   Type;
        UWORD   Count;
        APTR    ColorTable;
        struct  ViewPortExtra *cm_vpe;
        UWORD   *TransparencyBits;
        UBYTE   TransparencyPlane;
        UBYTE   reserved1;
        UWORD   reserved2;
        struct  ViewPort *cm_vp;
        APTR    NormalDisplayInfo;
        APTR    CoerceDisplayInfo;
        struct  TagItem *cm_batch_items;
        ULONG   VPModeID;
};

/* if Type == 0 then ColorMap is V1.2/V1.3 compatible */
```

```
/* if Type != 0 then ColorMap is V36       compatible */

#define COLORMAP_TYPE_V1_2      0x00
#define COLORMAP_TYPE_V1_4      0x01
#define COLORMAP_TYPE_V36 COLORMAP_TYPE_V1_4    /* use this definition */

/* Flags variable */
#define COLORMAP_TRANSPARENCY   0x01
#define COLORPLANE_TRANSPARENCY 0x02
#define BORDER_BLANKING         0x04
#define BORDER_NOTRANSPARENCY   0x08
#define VIDEOCONTROL_BATCH      0x10
#define USER_COPPER_CLIP        0x20

#endif  /* GRAPHICS_VIEW_H */
```

hardware/adkbits.h

```
#ifndef HARDWARE_ADKBITS_H
#define HARDWARE_ADKBITS_H
/*
**      $Filename: hardware/adkbits.h $
**      $Release: 2.04 $
**      $Revision: 36.2 $
**      $Date: 90/07/10 $
**
**      bit definitions for adkcon register
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#define  ADKB_SETCLR    15 /* standard set/clear bit */
#define  ADKB_PRECOMP1  14 /* two bits of precompensation */
#define  ADKB_PRECOMP0  13
#define  ADKB_MFMPREC   12 /* use mfm style precompensation */
#define  ADKB_UARTBRK   11 /* force uart output to zero */
#define  ADKB_WORDSYNC  10 /* enable DSKSYNC register matching */
#define  ADKB_MSBSYNC   9  /* (Apple GCR Only) sync on MSB for reading */
#define  ADKB_FAST      8  /* 1 -> 2 us/bit (mfm), 2 -> 4 us/bit (gcr) */
#define  ADKB_USE3PN    7  /* use aud chan 3 to modulate period of ?? */
#define  ADKB_USE2P3    6  /* use aud chan 2 to modulate period of 3 */
#define  ADKB_USE1P2    5  /* use aud chan 1 to modulate period of 2 */
#define  ADKB_USE0P1    4  /* use aud chan 0 to modulate period of 1 */
#define  ADKB_USE3VN    3  /* use aud chan 3 to modulate volume of ?? */
#define  ADKB_USE2V3    2  /* use aud chan 2 to modulate volume of 3 */
#define  ADKB_USE1V2    1  /* use aud chan 1 to modulate volume of 2 */
#define  ADKB_USE0V1    0  /* use aud chan 0 to modulate volume of 1 */

#define  ADKF_SETCLR    (1<<15)
#define  ADKF_PRECOMP1  (1<<14)
#define  ADKF_PRECOMP0  (1<<13)
#define  ADKF_MFMPREC   (1<<12)
#define  ADKF_UARTBRK   (1<<11)
#define  ADKF_WORDSYNC  (1<<10)
#define  ADKF_MSBSYNC   (1<<9)
#define  ADKF_FAST      (1<<8)
#define  ADKF_USE3PN    (1<<7)
#define  ADKF_USE2P3    (1<<6)
#define  ADKF_USE1P2    (1<<5)
#define  ADKF_USE0P1    (1<<4)
#define  ADKF_USE3VN    (1<<3)
#define  ADKF_USE2V3    (1<<2)
#define  ADKF_USE1V2    (1<<1)
#define  ADKF_USE0V1    (1<<0)

#define ADKF_PRE000NS   0                        /* 000 ns of precomp */
#define ADKF_PRE140NS   (ADKF_PRECOMP0) /* 140 ns of precomp */
#define ADKF_PRE280NS   (ADKF_PRECOMP1) /* 280 ns of precomp */
#define ADKF_PRE560NS   (ADKF_PRECOMP0|ADKF_PRECOMP1) /* 560 ns of precomp */

#endif  /* HARDWARE_ADKBITS_H */
```

hardware/blit.h

```
#ifndef HARDWARE_BLIT_H
#define HARDWARE_BLIT_H
/*
**      $Filename: hardware/blit.h $
**      $Release: 2.04 $
**      $Revision: 36.3 $
**      $Date: 90/11/05 $
**
**      Defines for direct hardware use of the blitter.
**
**      (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#define HSIZEBITS 6
#define VSIZEBITS 16-HSIZEBITS
#define HSIZEMASK 0x3f        /* 2^6 -- 1 */
#define VSIZEMASK 0x3FF       /* 2^10 - 1 */

/* all agnii support horizontal blit of at least 1024 bits (128 bytes) wide */
/* some agnii support horizontal blit of up to 32768 bits (4096 bytes) wide */

#ifndef  NO_BIG_BLITS
#define  MINBYTESPERROW 128
#define  MAXBYTESPERROW 4096
#else
#define  MAXBYTESPERROW 128
#endif

/* definitions for blitter control register 0 */

#define ABC    0x80
#define ABNC   0x40
#define ANBC   0x20
#define ANBNC  0x10
#define NABC   0x8
#define NABNC  0x4
#define NANBC  0x2
#define NANBNC 0x1

/* some commonly used operations */
#define A_OR_B    ABC|ANBC|NABC | ABNC|ANBNC|NABNC
#define A_OR_C    ABC|NABC|ABNC | ANBC|NANBC|ANBNC
#define A_XOR_C   NABC|ABNC   | NANBC|ANBNC
#define A_TO_D    ABC|ANBC|ABNC|ANBNC

#define BC0B_DEST 8
#define BC0B_SRCC 9
#define BC0B_SRCB   10
#define BC0B_SRCA 11
#define BC0F_DEST 0x100
#define BC0F_SRCC 0x200
#define BC0F_SRCB 0x400
#define BC0F_SRCA 0x800

#define BC1F_DESC   2         /* blitter descend direction */

#define DEST 0x100
#define SRCC 0x200
#define SRCB 0x400
#define SRCA 0x800

#define ASHIFTSHIFT  12       /* bits to right align ashift value */
#define BSHIFTSHIFT  12       /* bits to right align bshift value */

/* definations for blitter control register 1 */
```

hardware/blit.h Page 2

```
#define LINEMODE     0x1
#define FILL_OR      0x8
#define FILL_XOR     0x10
#define FILL_CARRYIN 0x4
#define ONEDOT       0x2     /* one dot per horizontal line */
#define OVFLAG       0x20
#define SIGNFLAG     0x40
#define BLITREVERSE  0x2

#define SUD          0x10
#define SUL          0x8
#define AUL          0x4

#define OCTANT8   24
#define OCTANT7   4
#define OCTANT6   12
#define OCTANT5   28
#define OCTANT4   20
#define OCTANT3   8
#define OCTANT2   0
#define OCTANT1   16

/* stuff for blit qeuer */
struct bltnode
{
    struct  bltnode *n;
    int     (*function)();
    char    stat;
    short   blitsize;
    short   beamsync;
    int     (*cleanup)();
};

/* defined bits for bltstat */
#define CLEANUP 0x40
#define CLEANME CLEANUP

#endif  /* HARDWARE_BLIT_H */
```

hardware/cia.h Page 1

```
#ifndef HARDWARE_CIA_H
#define HARDWARE_CIA_H
/*
**      $Filename: hardware/cia.h $
**      $Release: 2.04 $
**      $Revision: 36.3 $
**      $Date: 90/11/05 $
**
**      registers and bits in the Complex Interface Adapter (CIA) chip
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/


#ifndef EXEC_TYPES_H
#include "exec/types.h"
#endif /* EXEC_TYPES_H */



/*
 * ciaa is on an ODD address (e.g. the low byte) -- $bfe001
 * ciab is on an EVEN address (e.g. the high byte) -- $bfd000
 *
 * do this to get the definitions:
 *    extern struct CIA ciaa, ciab;
 */


struct CIA {
    UBYTE   ciapra;
    UBYTE   pad0[0xff];
    UBYTE   ciaprb;
    UBYTE   pad1[0xff];
    UBYTE   ciaddra;
    UBYTE   pad2[0xff];
    UBYTE   ciaddrb;
    UBYTE   pad3[0xff];
    UBYTE   ciatalo;
    UBYTE   pad4[0xff];
    UBYTE   ciatahi;
    UBYTE   pad5[0xff];
    UBYTE   ciatblo;
    UBYTE   pad6[0xff];
    UBYTE   ciatbhi;
    UBYTE   pad7[0xff];
    UBYTE   ciatodlow;
    UBYTE   pad8[0xff];
    UBYTE   ciatodmid;
    UBYTE   pad9[0xff];
    UBYTE   ciatodhi;
    UBYTE   pad10[0xff];
    UBYTE   unusedreg;
    UBYTE   pad11[0xff];
    UBYTE   ciasdr;
    UBYTE   pad12[0xff];
    UBYTE   ciaicr;
    UBYTE   pad13[0xff];
    UBYTE   ciacra;
    UBYTE   pad14[0xff];
    UBYTE   ciacrb;
};


/* interrupt control register bit numbers */
```

```
#define CIAICRB_TA      0
#define CIAICRB_TB      1
#define CIAICRB_ALRM    2
#define CIAICRB_SP      3
#define CIAICRB_FLG     4
#define CIAICRB_IR      7
#define CIAICRB_SETCLR  7

/* control register A bit numbers */
#define CIACRAB_START   0
#define CIACRAB_PBON    1
#define CIACRAB_OUTMODE 2
#define CIACRAB_RUNMODE 3
#define CIACRAB_LOAD    4
#define CIACRAB_INMODE  5
#define CIACRAB_SPMODE  6
#define CIACRAB_TODIN   7

/* control register B bit numbers */
#define CIACRBB_START   0
#define CIACRBB_PBON    1
#define CIACRBB_OUTMODE 2
#define CIACRBB_RUNMODE 3
#define CIACRBB_LOAD    4
#define CIACRBB_INMODE0 5
#define CIACRBB_INMODE1 6
#define CIACRBB_ALARM   7

/* interrupt control register masks */
#define CIAICRF_TA      (1<<CIAICRB_TA)
#define CIAICRF_TB      (1<<CIAICRB_TB)
#define CIAICRF_ALRM    (1<<CIAICRB_ALRM)
#define CIAICRF_SP      (1<<CIAICRB_SP)
#define CIAICRF_FLG     (1<<CIAICRB_FLG)
#define CIAICRF_IR      (1<<CIAICRB_IR)
#define CIAICRF_SETCLR  (1<<CIAICRB_SETCLR)

/* control register A register masks */
#define CIACRAF_START   (1<<CIACRAB_START)
#define CIACRAF_PBON    (1<<CIACRAB_PBON)
#define CIACRAF_OUTMODE (1<<CIACRAB_OUTMODE)
#define CIACRAF_RUNMODE (1<<CIACRAB_RUNMODE)
#define CIACRAF_LOAD    (1<<CIACRAB_LOAD)
#define CIACRAF_INMODE  (1<<CIACRAB_INMODE)
#define CIACRAF_SPMODE  (1<<CIACRAB_SPMODE)
#define CIACRAF_TODIN   (1<<CIACRAB_TODIN)

/* control register B register masks */
#define CIACRBF_START   (1<<CIACRBB_START)
#define CIACRBF_PBON    (1<<CIACRBB_PBON)
#define CIACRBF_OUTMODE (1<<CIACRBB_OUTMODE)
#define CIACRBF_RUNMODE (1<<CIACRBB_RUNMODE)
#define CIACRBF_LOAD    (1<<CIACRBB_LOAD)
#define CIACRBF_INMODE0 (1<<CIACRBB_INMODE0)
#define CIACRBF_INMODE1 (1<<CIACRBB_INMODE1)
#define CIACRBF_ALARM   (1<<CIACRBB_ALARM)

/* control register B INMODE masks */
#define CIACRBF_IN_PHI2 0
#define CIACRBF_IN_CNT  (CIACRBF_INMODE0)
#define CIACRBF_IN_TA   (CIACRBF_INMODE1)
#define CIACRBF_IN_CNT_TA  (CIACRBF_INMODE0|CIACRBF_INMODE1)

/*
 * Port definitions -- what each bit in a cia peripheral register is tied to
 */
```

```

/* ciaa port A (0xbfe001) */
#define CIAB_GAMEPORT1  (7)   /* gameport 1, pin 6 (fire button*) */
#define CIAB_GAMEPORT0  (6)   /* gameport 0, pin 6 (fire button*) */
#define CIAB_DSKRDY     (5)   /* disk ready* */
#define CIAB_DSKTRACK0  (4)   /* disk on track 00* */
#define CIAB_DSKPROT    (3)   /* disk write protect* */
#define CIAB_DSKCHANGE  (2)   /* disk change* */
#define CIAB_LED        (1)   /* led light control (0==>bright) */
#define CIAB_OVERLAY    (0)   /* memory overlay bit */

/* ciaa port B (0xbfe101) -- parallel port */

/* ciab port A (0xbfd000) -- serial and printer control */
#define CIAB_COMDTR     (7)   /* serial Data Terminal Ready* */
#define CIAB_COMRTS     (6)   /* serial Request to Send* */
#define CIAB_COMCD      (5)   /* serial Carrier Detect* */
#define CIAB_COMCTS     (4)   /* serial Clear to Send* */
#define CIAB_COMDSR     (3)   /* serial Data Set Ready* */
#define CIAB_PRTRSEL    (2)   /* printer SELECT */
#define CIAB_PRTRPOUT   (1)   /* printer paper out */
#define CIAB_PRTRBUSY   (0)   /* printer busy */

/* ciab port B (0xbfd100) -- disk control */
#define CIAB_DSKMOTOR   (7)   /* disk motorr* */
#define CIAB_DSKSEL3    (6)   /* disk select unit 3* */
#define CIAB_DSKSEL2    (5)   /* disk select unit 2* */
#define CIAB_DSKSEL1    (4)   /* disk select unit 1* */
#define CIAB_DSKSEL0    (3)   /* disk select unit 0* */
#define CIAB_DSKSIDE    (2)   /* disk side select* */
#define CIAB_DSKDIREC   (1)   /* disk direction of seek* */
#define CIAB_DSKSTEP    (0)   /* disk step heads* */

/* ciaa port A (0xbfe001) */
#define CIAF_GAMEPORT1  (1<<7)
#define CIAF_GAMEPORT0  (1<<6)
#define CIAF_DSKRDY     (1<<5)
#define CIAF_DSKTRACK0  (1<<4)
#define CIAF_DSKPROT    (1<<3)
#define CIAF_DSKCHANGE  (1<<2)
#define CIAF_LED        (1<<1)
#define CIAF_OVERLAY    (1<<0)

/* ciaa port B (0xbfe101) -- parallel port */

/* ciab port A (0xbfd000) -- serial and printer control */
#define CIAF_COMDTR     (1<<7)
#define CIAF_COMRTS     (1<<6)
#define CIAF_COMCD      (1<<5)
#define CIAF_COMCTS     (1<<4)
#define CIAF_COMDSR     (1<<3)
#define CIAF_PRTRSEL    (1<<2)
#define CIAF_PRTRPOUT   (1<<1)
#define CIAF_PRTRBUSY   (1<<0)

/* ciab port B (0xbfd100) -- disk control */
#define CIAF_DSKMOTOR   (1<<7)
#define CIAF_DSKSEL3    (1<<6)
#define CIAF_DSKSEL2    (1<<5)
#define CIAF_DSKSEL1    (1<<4)
#define CIAF_DSKSEL0    (1<<3)
#define CIAF_DSKSIDE    (1<<2)
#define CIAF_DSKDIREC   (1<<1)
#define CIAF_DSKSTEP    (1<<0)

#endif  /* HARDWARE_CIA_H */
```

hardware/custom.h

```
#ifndef HARDWARE_CUSTOM_H
#define HARDWARE_CUSTOM_H
/*
**      $Filename: hardware/custom.h $
**      $Release: 2.04 $
**      $Revision: 36.4 $
**      $Date: 90/11/05 $
**
**      Offsets of Amiga custom chip registers
**
**      (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_TYPES_H
#include "exec/types.h"
#endif /* EXEC_TYPES_H */



/*
 * do this to get base of custom registers:
 * extern struct Custom custom;
 */


struct Custom {
    UWORD   bltddat;
    UWORD   dmaconr;
    UWORD   vposr;
    UWORD   vhposr;
    UWORD   dskdatr;
    UWORD   joy0dat;
    UWORD   joy1dat;
    UWORD   clxdat;
    UWORD   adkconr;
    UWORD   pot0dat;
    UWORD   pot1dat;
    UWORD   potinp;
    UWORD   serdatr;
    UWORD   dskbytr;
    UWORD   intenar;
    UWORD   intreqr;
    APTR    dskpt;
    UWORD   dsklen;
    UWORD   dskdat;
    UWORD   refptr;
    UWORD   vposw;
    UWORD   vhposw;
    UWORD   copcon;
    UWORD   serdat;
    UWORD   serper;
    UWORD   potgo;
    UWORD   joytest;
    UWORD   strequ;
    UWORD   strvbl;
    UWORD   strhor;
    UWORD   strlong;
    UWORD   bltcon0;
    UWORD   bltcon1;
    UWORD   bltafwm;
    UWORD   bltalwm;
    APTR    bltcpt;
    APTR    bltbpt;
    APTR    bltapt;
    APTR    bltdpt;
```

hardware/custom.h

```
    UWORD   bltsize;
    UBYTE   pad2d;
    UBYTE   bltcon0l;   /* low 8 bits of bltcon0, write only */
    UWORD   bltsizv;
    UWORD   bltsizh;    /* 5e */
    UWORD   bltcmod;
    UWORD   bltbmod;
    UWORD   bltamod;
    UWORD   bltdmod;
    UWORD   pad34[4];
    UWORD   bltcdat;
    UWORD   bltbdat;
    UWORD   bltadat;
    UWORD   pad3b[3];
    UWORD deniseid;   /* 7c */
    UWORD   dsksync;
    ULONG   cop1lc;
    ULONG   cop2lc;
    UWORD   copjmp1;
    UWORD   copjmp2;
    UWORD   copins;
    UWORD   diwstrt;
    UWORD   diwstop;
    UWORD   ddfstrt;
    UWORD   ddfstop;
    UWORD   dmacon;
    UWORD   clxcon;
    UWORD   intena;
    UWORD   intreq;
    UWORD   adkcon;
    struct  AudChannel {
      UWORD *ac_ptr; /* ptr to start of waveform data */
      UWORD ac_len;     /* length of waveform in words */
      UWORD ac_per;     /* sample period */
      UWORD ac_vol;     /* volume */
      UWORD ac_dat;     /* sample pair */
      UWORD ac_pad[2];  /* unused */
    } aud[4];
    APTR    bplpt[8];
    UWORD   bplcon0;
    UWORD   bplcon1;
    UWORD   bplcon2;
    UWORD   bplcon3;
    UWORD   bpl1mod;
    UWORD   bpl2mod;
    UWORD   bplhmod;
    UWORD   pad86[1];
    UWORD   bpldat[8];
    APTR    sprpt[8];
    struct  SpriteDef {
      UWORD pos;
      UWORD ctl;
      UWORD dataa;
      UWORD datab;
    } spr[8];
    UWORD   color[32];
    UWORD htotal;
    UWORD hsstop;
    UWORD hbstrt;
    UWORD hbstop;
    UWORD vtotal;
    UWORD vsstop;
    UWORD vbstrt;
    UWORD vbstop;
    UWORD sprhstrt;
    UWORD sprhstop;
```

hardware/custom.h

Page 3

```
    UWORD bplhstrt;
    UWORD bplhstop;
    UWORD hhposw;
    UWORD hhposr;
    UWORD beamcon0;
    UWORD hsstrt;
    UWORD vsstrt;
    UWORD hcenter;
    UWORD   diwhigh;    /* 1e4 */
};

#ifdef ECS_SPECIFIC

/* defines for beamcon register */
#define VARVBLANK       0x1000  /* Variable vertical blank enable */
#define LOLDIS          0x0800  /* long line disable */
#define CSCBLANKEN      0x0400  /* redirect composite sync */
#define VARVSYNC        0x0200  /* Variable vertical sync enable */
#define VARHSYNC        0x0100  /* Variable horizontal sync enable */
#define VARBEAM 0x0080  /* variable beam counter enable */
#define DISPLAYDUAL     0x0040  /* use UHRES pointer and standard pointers */
#define DISPLAYPAL      0x0020  /* set decodes to generate PAL display */
#define VARCSYNC        0x0010  /* Variable composite sync enable */
#define CSBLANK 0x0008  /* Composite blank out to CSY* pin */
#define CSYNCTRUE       0x0004  /* composite sync true signal */
#define VSYNCTRUE       0x0002  /* vertical sync true */
#define HSYNCTRUE       0x0001  /* horizontal sync true */

/* new defines for bplcon0 */
#define USE_BPLCON3     1

/* new defines for bplcon2 */
#define BPLCON2_ZDCTEN          (1<<10) /* colormapped genlock bit */
#define BPLCON2_ZDBPEN          (1<<11) /* use bitplane as genlock bits */
#define BPLCON2_ZDBPSEL0        (1<<12) /* three bits to select one */
#define BPLCON2_ZDBPSEL1        (1<<13) /* of 8 bitplanes in */
#define BPLCON2_ZDBPSEL2        (1<<14) /* ZDBPEN genlock mode */

/* defines for bplcon3 register */
#define BPLCON3_EXTBLNKEN       (1<<0)  /* external blank enable */
#define BPLCON3_EXTBLKZD        (1<<1)  /* external blank ored into trnsprncy */
#define BPLCON3_ZDCLKEN (1<<2)  /* zd pin outputs a 14mhz clock*/
#define BPLCON3_BRDNTRAN        (1<<4)  /* border is opaque */
#define BPLCON3_BRDNBLNK        (1<<5)  /* border is opaque */

#endif  /* ECS_SPECIFIC */

#endif  /* HARDWARE_CUSTOM_H */
```

hardware/dmabits.h

Page 1

```
#ifndef HARDWARE_DMABITS_H
#define HARDWARE_DMABITS_H
/*
**      $Filename: hardware/dmabits.h $
**      $Release: 2.04 $
**      $Revision: 36.2 $
**      $Date: 90/07/10 $
**
**      include file for defining dma control stuff
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

/* write definitions for dmaconw */
#define DMAF_SETCLR  0x8000
#define DMAF_AUDIO   0x000F   /* 4 bit mask */
#define DMAF_AUD0    0x0001
#define DMAF_AUD1    0x0002
#define DMAF_AUD2    0x0004
#define DMAF_AUD3    0x0008
#define DMAF_DISK    0x0010
#define DMAF_SPRITE  0x0020
#define DMAF_BLITTER 0x0040
#define DMAF_COPPER  0x0080
#define DMAF_RASTER  0x0100
#define DMAF_MASTER  0x0200
#define DMAF_BLITHOG 0x0400
#define DMAF_ALL     0x01FF   /* all dma channels */

/* read definitions for dmaconr */
/* bits 0-8 correspnd to dmaconw definitions */
#define DMAF_BLTDONE 0x4000
#define DMAF_BLTNZERO   0x2000

#define DMAB_SETCLR  15
#define DMAB_AUD0    0
#define DMAB_AUD1    1
#define DMAB_AUD2    2
#define DMAB_AUD3    3
#define DMAB_DISK    4
#define DMAB_SPRITE  5
#define DMAB_BLITTER 6
#define DMAB_COPPER  7
#define DMAB_RASTER  8
#define DMAB_MASTER  9
#define DMAB_BLITHOG 10
#define DMAB_BLTDONE 14
#define DMAB_BLTNZERO   13

#endif  /* HARDWARE_DMABITS_H */
```

hardware/intbits.h

```
#ifndef HARDWARE_INTBITS_H
#define HARDWARE_INTBITS_H
/*
**      $Filename: hardware/intbits.h $
**      $Release: 2.04 $
**      $Revision: 36.2 $
**      $Date: 90/07/10 $
**
**      bits in the interrupt enable (and interrupt request) register
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#define  INTB_SETCLR    (15)  /* Set/Clear control bit. Determines if bits */
            /* written with a 1 get set or cleared. Bits */
            /* written with a zero are allways unchanged */
#define  INTB_INTEN     (14)  /* Master interrupt (enable only ) */
#define  INTB_EXTER     (13)  /* External interrupt */
#define  INTB_DSKSYNC   (12)  /* Disk re-SYNChronized */
#define  INTB_RBF       (11)  /* serial port Receive Buffer Full */
#define  INTB_AUD3      (10)  /* Audio channel 3 block finished */
#define  INTB_AUD2      (9)   /* Audio channel 2 block finished */
#define  INTB_AUD1      (8)   /* Audio channel 1 block finished */
#define  INTB_AUD0      (7)   /* Audio channel 0 block finished */
#define  INTB_BLIT      (6)   /* Blitter finished */
#define  INTB_VERTB     (5)   /* start of Vertical Blank */
#define  INTB_COPER     (4)   /* Coprocessor */
#define  INTB_PORTS     (3)   /* I/O Ports and timers */
#define  INTB_SOFTINT   (2)   /* software interrupt request */
#define  INTB_DSKBLK    (1)   /* Disk Block done */
#define  INTB_TBE       (0)   /* serial port Transmit Buffer Empty */



#define  INTF_SETCLR    (1<<15)
#define  INTF_INTEN     (1<<14)
#define  INTF_EXTER     (1<<13)
#define  INTF_DSKSYNC   (1<<12)
#define  INTF_RBF       (1<<11)
#define  INTF_AUD3      (1<<10)
#define  INTF_AUD2      (1<<9)
#define  INTF_AUD1      (1<<8)
#define  INTF_AUD0      (1<<7)
#define  INTF_BLIT      (1<<6)
#define  INTF_VERTB     (1<<5)
#define  INTF_COPER     (1<<4)
#define  INTF_PORTS     (1<<3)
#define  INTF_SOFTINT   (1<<2)
#define  INTF_DSKBLK    (1<<1)
#define  INTF_TBE       (1<<0)

#endif  /* HARDWARE_INTBITS_H */
```

```
#ifndef INTUITION_CGHOOKS_H
#define INTUITION_CGHOOKS_H 1
/*
**  $Filename: intuition/cghooks.h $
**  $Release: 2.04 $
**  $Revision: 36.1 $
**  $Date: 90/11/01 $
**
**  Custom Gadget processing
**
**  (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_TYPES_H
#include <exec/types.h>
#endif

#ifndef INTUITION_INTUITION_H
#include <intuition/intuition.h>
#endif

/*
 * Package of information passed to custom and 'boopsi'
 * gadget "hook" functions.  This structure is READ ONLY.
 */
struct GadgetInfo {

    struct Screen               *gi_Screen;
    struct Window               *gi_Window;     /* null for screen gadgets */
    struct Requester            *gi_Requester;  /* null if not GTYP_REQGADGET */

    /* rendering information:
     * don't use these without cloning/locking.
     * Official way is to call ObtainRPort()
     */
    struct RastPort             *gi_RastPort;
    struct Layer                *gi_Layer;

    /* copy of dimensions of screen/window/g00/req(/group)
     * that gadget resides in.  Left/Top of this box is
     * offset from window mouse coordinates to gadget coordinates
     *          screen gadgets:             0,0 (from screen coords)
     *  window gadgets (no g00):        0,0
     *  GTYP_GZZGADGETs (borderlayer):          0,0
     *  GZZ innerlayer gadget:          borderleft, bordertop
     *  Requester gadgets:              reqleft, reqtop
     */
    struct IBox                 gi_Domain;

    /* these are the pens for the window or screen      */
    struct {
        UBYTE   DetailPen;
        UBYTE   BlockPen;
    }                           gi_Pens;

    /* the Detail and Block pens in gi_DrInfo->dri_Pens[] are
     * for the screen.  Use the above for window-sensitive
     * colors.
     */
    struct DrawInfo             *gi_DrInfo;

    /* reserved space: this structure is extensible
     * anyway, but using these saves some recompilation
     */
    ULONG                       gi_Reserved[6];
```

```
};

/*** system private data structure for now ***/
/* prop gadget extra info       */
struct PGX      {
    struct IBox pgx_Container;
    struct IBox pgx_NewKnob;
};

/* this casts MutualExclude for easy assignment of a hook
 * pointer to the unused MutualExclude field of a custom gadget
 */
#define CUSTOM_HOOK( gadget ) ( (struct Hook *) (gadget)->MutualExclude)

#endif
```

intuition/classes.h Page 1

```
#ifndef INTUITION_CLASSES_H
#define INTUITION_CLASSES_H     1
/*
**  $Filename: intuition/classes.h $
**  $Release: 2.04 $
**  $Revision: 36.1 $
**  $Date: 90/11/01 $
**
**  Used only by class implementors
**
**  (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef UTILITY_HOOKS_H
#include <utility/hooks.h>
#endif

#ifndef INTUITION_CLASSUSR_H
#include <intuition/classusr.h>
#endif

/*******************************************/
/*** "White box" access to struct IClass ***/
/*******************************************/

/* This structure is READ-ONLY, and allocated only by Intuition */
typedef struct IClass {
    struct Hook         cl_Dispatcher;
    ULONG               cl_Reserved;    /* must be 0  */
    struct IClass       *cl_Super;
    ClassID             cl_ID;

    /* where within an object is the instance data for this class? */
    UWORD               cl_InstOffset;
    UWORD               cl_InstSize;

    ULONG               cl_UserData;    /* per-class data of your choice */
    ULONG               cl_SubclassCount;
                                        /* how many direct subclasses?  */
    ULONG               cl_ObjectCount;
                                /* how many objects created of this class? */
    ULONG               cl_Flags;
#define CLF_INLIST      0x00000001      /* class is in public class list */
} Class;

/* add offset for instance data to an object handle */
#define INST_DATA( cl, o )      ((VOID *) (((UBYTE *)o)+cl->cl_InstOffset))

/* sizeof the instance data for a given class */
#define SIZEOF_INSTANCE( cl )   ((cl)->cl_InstOffset + (cl)->cl_InstSize \
                        + sizeof (struct _Object ))

/**************************************************/
/*** "White box" access to struct _Object       ***/
/**************************************************/

/*
 * We have this, the instance data of the root class, PRECEDING
 * the "object".  This is so that Gadget objects are Gadget pointers,
 * and so on.  If this structure grows, it will always have o_Class
 * at the end, so the macro OCLASS(o) will always have the same
 * offset back from the pointer returned from NewObject().
 *
 * This data structure is subject to change.  Do not use the o_Node
 * embedded structure.
```

intuition/classes.h Page 2

```
 */
struct _Object {
    struct MinNode      o_Node;
    struct IClass       *o_Class;
};

/* convenient typecast  */
#define _OBJ( o )       ((struct _Object *)(o))

/* get "public" handle on baseclass instance from real beginning of obj data */
#define BASEOBJECT( _obj )      ( (Object *) (_OBJ(_obj)+1) )

/* get back to object data struct from public handle */
#define _OBJECT( o )            (_OBJ(o) - 1)

/* get class pointer from an object handle      */
#define OCLASS( o )     ( (_OBJECT(o))->o_Class )

#endif
```

intuition/classusr.h

```
#ifndef INTUITION_CLASSUSR_H
#define INTUITION_CLASSUSR_H    1
/*
**  $Filename: intuition/classusr.h $
**  $Release: 2.04 $
**  $Revision: 36.1 $
**  $Date: 90/11/01 $
**
**  For application users of Intuition object classes
**
**  (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/


#ifndef UTILITY_HOOKS_H
#include <utility/hooks.h>
#endif

/*** User visible handles on objects, classes, messages ***/
typedef ULONG   Object;         /* abstract handle */

typedef UBYTE   *ClassID;

/* you can use this type to point to a "generic" message,
 * in the object-oriented programming parlance.  Based on
 * the value of 'MethodID', you dispatch to processing
 * for the various message types.  The meaningful parameter
 * packet structure definitions are defined below.
 */
typedef struct {
    ULONG MethodID;
    /* method-specific data follows, some examples below */
}               *Msg;

/*
 * Class id strings for Intuition classes.
 * There's no real reason to use the uppercase constants
 * over the lowercase strings, but this makes a good place
 * to list the names of the built-in classes.
 */
#define ROOTCLASS       "rootclass"             /* classusr.h   */
#define IMAGECLASS      "imageclass"            /* imageclass.h */
#define FRAMEICLASS     "frameiclass"
#define SYSICLASS       "sysiclass"
#define FILLRECTCLASS   "fillrectclass"
#define GADGETCLASS     "gadgetclass"           /* gadgetclass.h */
#define PROPGCLASS      "propgclass"
#define STRGCLASS       "strgclass"
#define BUTTONGCLASS    "buttongclass"
#define FRBUTTONCLASS   "frbuttonclass"
#define GROUPGCLASS     "groupgclass"
#define ICCLASS         "icclass"               /* icclass.h    */
#define MODELCLASS      "modelclass"


/* Dispatched method ID's
 * NOTE: Applications should use Intuition entry points, not direct
 * DoMethod() calls, for NewObject, DisposeObject, SetAttrs,
 * SetGadgetAttrs, and GetAttr.
 */

#define OM_Dummy        (0x100)
#define OM_NEW          (0x101) /* 'object' parameter is "true class"   */
#define OM_DISPOSE      (0x102) /* delete self (no parameters)          */
#define OM_SET          (0x103) /* set attributes (in tag list)         */
```

```
#define OM_GET          (0x104) /* return single attribute value        */
#define OM_ADDTAIL      (0x105) /* add self to a List (let root do it)  */
#define OM_REMOVE       (0x106) /* remove self from list                */
#define OM_NOTIFY       (0x107) /* send to self: notify dependents      */
#define OM_UPDATE       (0x108) /* notification message from somebody   */
#define OM_ADDMEMBER    (0x109) /* used by various classes with lists   */
#define OM_REMMEMBER    (0x10A) /* used by various classes with lists   */

/* Parameter "Messages" passed to methods       */

/* OM_NEW and OM_SET    */
struct opSet {
    ULONG               MethodID;
    struct TagItem      *ops_AttrList;  /* new attributes       */
    struct GadgetInfo   *ops_GInfo;     /* always there for gadgets,
                                         * when SetGadgetAttrs() is used,
                                         * but will be NULL for OM_NEW
                                         */
};

/* OM_NOTIFY, and OM_UPDATE     */
struct opUpdate {
    ULONG               MethodID;
    struct TagItem      *opu_AttrList;  /* new attributes       */
    struct GadgetInfo   *opu_GInfo;     /* non-NULL when SetGadgetAttrs or
                                         * notification resulting from gadget
                                         * input occurs.
                                         */
    ULONG               opu_Flags;      /* defined below        */
};

/* this flag means that the update message is being issued from
 * something like an active gadget, a la GACT_FOLLOWMOUSE.  When
 * the gadget goes inactive, it will issue a final update
 * message with this bit cleared.  Examples of use are for
 * GACT_FOLLOWMOUSE equivalents for propgadclass, and repeat strobes
 * for buttons.
 */
#define OPUF_INTERIM    (1<<0)

/* OM_GET       */
struct opGet {
    ULONG               MethodID;
    ULONG               opg_AttrID;
    ULONG               *opg_Storage;   /* may be other types, but "int"
                                         * types are all ULONG
                                         */
};

/* OM_ADDTAIL   */
struct opAddTail {
    ULONG               MethodID;
    struct List         *opat_List;
};

/* OM_ADDMEMBER, OM_REMMEMBER   */
#define  opAddMember opMember
struct opMember {
    ULONG               MethodID;
    Object              *opam_Object;
};


#endif
```

```
#ifndef INTUITION_GADGETCLASS_H
#define INTUITION_GADGETCLASS_H 1
/*
**  $Filename: intuition/gadgetclass.h $
**  $Release: 2.04 $
**  $Revision: 36.5 $
**  $Date: 91/02/22 $
**
**  Custom and 'boopsi' gadget class interface
**
**  (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_TYPES_H
#include <exec/types.h>
#endif

#ifndef INTUITION_INTUITION_H
#include <intuition/intuition.h>
#endif

#ifndef UTILITY_TAGITEM_H
#include <utility/tagitem.h>
#endif

/*
 * NOTE:  <intuition/iobsolete.h> is included at the END of this file!
 */

/* Gadget Class attributes      */

#define    GA_Dummy              (TAG_USER +0x30000)
#define    GA_Left               (GA_Dummy + 0x0001)
#define    GA_RelRight           (GA_Dummy + 0x0002)
#define    GA_Top                (GA_Dummy + 0x0003)
#define    GA_RelBottom          (GA_Dummy + 0x0004)
#define    GA_Width              (GA_Dummy + 0x0005)
#define    GA_RelWidth           (GA_Dummy + 0x0006)
#define    GA_Height             (GA_Dummy + 0x0007)
#define    GA_RelHeight          (GA_Dummy + 0x0008)
#define    GA_Text               (GA_Dummy + 0x0009)  /* ti_Data is (UBYTE *) */
#define    GA_Image              (GA_Dummy + 0x000A)
#define    GA_Border             (GA_Dummy + 0x000B)
#define    GA_SelectRender       (GA_Dummy + 0x000C)
#define    GA_Highlight          (GA_Dummy + 0x000D)
#define    GA_Disabled           (GA_Dummy + 0x000E)
#define    GA_GZZGadget          (GA_Dummy + 0x000F)
#define    GA_ID                 (GA_Dummy + 0x0010)
#define    GA_UserData           (GA_Dummy + 0x0011)
#define    GA_SpecialInfo        (GA_Dummy + 0x0012)
#define    GA_Selected           (GA_Dummy + 0x0013)
#define    GA_EndGadget          (GA_Dummy + 0x0014)
#define    GA_Immediate          (GA_Dummy + 0x0015)
#define    GA_RelVerify          (GA_Dummy + 0x0016)
#define    GA_FollowMouse        (GA_Dummy + 0x0017)
#define    GA_RightBorder        (GA_Dummy + 0x0018)
#define    GA_LeftBorder         (GA_Dummy + 0x0019)
#define    GA_TopBorder          (GA_Dummy + 0x001A)
#define    GA_BottomBorder       (GA_Dummy + 0x001B)
#define    GA_ToggleSelect       (GA_Dummy + 0x001C)

    /* internal use only, until further notice, please */
#define    GA_SysGadget          (GA_Dummy + 0x001D)
        /* bool, sets GTYP_SYSGADGET field in type      */
#define    GA_SysGType           (GA_Dummy + 0x001E)
```

```
        /* e.g., GTYP_WUPFRONT, ...     */

#define    GA_Previous           (GA_Dummy + 0x001F)
        /* previous gadget (or (struct Gadget **)) in linked list
         * NOTE: This attribute CANNOT be used to link new gadgets
         * into the gadget list of an open window or requester.
         * You must use AddGList().
         */

#define    GA_Next               (GA_Dummy + 0x0020)
         /* not implemented */

#define    GA_DrawInfo           (GA_Dummy + 0x0021)
        /* some fancy gadgets need to see a DrawInfo
         * when created or for layout
         */

/* You should use at most ONE of GA_Text, GA_IntuiText, and GA_LabelImage */
#define GA_IntuiText             (GA_Dummy + 0x0022)
        /* ti_Data is (struct IntuiText *) */

#define GA_LabelImage            (GA_Dummy + 0x0023)
        /* ti_Data is an image (object), used in place of
         * GadgetText
         */

#define GA_TabCycle              (GA_Dummy + 0x0024)
        /* New for V37:
         * Boolean indicates that this gadget is to participate in
         * cycling activation with Tab or Shift-Tab.
         */

/* PROPGCLASS attributes */

#define PGA_Dummy       (TAG_USER + 0x31000)
#define PGA_Freedom     (PGA_Dummy + 0x0001)
        /* only one of FREEVERT or FREEHORIZ */
#define PGA_Borderless  (PGA_Dummy + 0x0002)
#define PGA_HorizPot    (PGA_Dummy + 0x0003)
#define PGA_HorizBody   (PGA_Dummy + 0x0004)
#define PGA_VertPot     (PGA_Dummy + 0x0005)
#define PGA_VertBody    (PGA_Dummy + 0x0006)
#define PGA_Total       (PGA_Dummy + 0x0007)
#define PGA_Visible     (PGA_Dummy + 0x0008)
#define PGA_Top         (PGA_Dummy + 0x0009)
/* New for V37: */
#define PGA_NewLook     (PGA_Dummy + 0x000A)

/* STRGCLASS attributes */

#define STRINGA_Dummy                   (TAG_USER      +0x32000)
#define STRINGA_MaxChars        (STRINGA_Dummy + 0x0001)
#define STRINGA_Buffer          (STRINGA_Dummy + 0x0002)
#define STRINGA_UndoBuffer      (STRINGA_Dummy + 0x0003)
#define STRINGA_WorkBuffer      (STRINGA_Dummy + 0x0004)
#define STRINGA_BufferPos       (STRINGA_Dummy + 0x0005)
#define STRINGA_DispPos         (STRINGA_Dummy + 0x0006)
#define STRINGA_AltKeyMap       (STRINGA_Dummy + 0x0007)
#define STRINGA_Font            (STRINGA_Dummy + 0x0008)
#define STRINGA_Pens            (STRINGA_Dummy + 0x0009)
#define STRINGA_ActivePens      (STRINGA_Dummy + 0x000A)
#define STRINGA_EditHook        (STRINGA_Dummy + 0x000B)
#define STRINGA_EditModes       (STRINGA_Dummy + 0x000C)

/* booleans */
#define STRINGA_ReplaceMode     (STRINGA_Dummy + 0x000D)
```

```
#define STRINGA_FixedFieldMode  (STRINGA_Dummy + 0x000E)
#define STRINGA_NoFilterMode    (STRINGA_Dummy + 0x000F)

#define STRINGA_Justification   (STRINGA_Dummy + 0x0010)
        /* GACT_STRINGCENTER, GACT_STRINGLEFT, GACT_STRINGRIGHT */
#define STRINGA_LongVal         (STRINGA_Dummy + 0x0011)
#define STRINGA_TextVal         (STRINGA_Dummy + 0x0012)

#define STRINGA_ExitHelp        (STRINGA_Dummy + 0x0013)
        /* STRINGA_ExitHelp is new for V37, and ignored by V36.
         * Set this if you want the gadget to exit when Help is
         * pressed.  Look for a code of 0x5F, the rawkey code for Help
         */

#define SG_DEFAULTMAXCHARS      (128)

/* Gadget Layout related attributes     */

#define LAYOUTA_Dummy           (TAG_USER  + 0x38000)
#define LAYOUTA_LayoutObj       (LAYOUTA_Dummy + 0x0001)
#define LAYOUTA_Spacing         (LAYOUTA_Dummy + 0x0002)
#define LAYOUTA_Orientation     (LAYOUTA_Dummy + 0x0003)

/* orientation values   */
#define LORIENT_NONE    0
#define LORIENT_HORIZ   1
#define LORIENT_VERT    2


/* Gadget Method ID's   */

#define GM_Dummy        (-1)    /* not used for anything                */
#define GM_HITTEST      (0)     /* return GMR_GADGETHIT if you are clicked on
                                 * (whether or not you are disabled).
                                 */
#define GM_RENDER       (1)     /* draw yourself, in the appropriate state */
#define GM_GOACTIVE     (2)     /* you are now going to be fed input    */
#define GM_HANDLEINPUT  (3)     /* handle that input                    */
#define GM_GOINACTIVE   (4)     /* whether or not by choice, you are done  */

/* Parameter "Messages" passed to gadget class methods  */

/* GM_HITTEST   */
struct gpHitTest {
    ULONG               MethodID;
    struct GadgetInfo   *gpht_GInfo;
    struct {
        WORD    X;
        WORD    Y;
    }                   gpht_Mouse;
};
/* GM_HITTEST return value */
#define GMR_GADGETHIT   (0x00000004)    /* if no hit, return 0 */

/* GM_RENDER    */
struct gpRender {
    ULONG               MethodID;
    struct GadgetInfo   *gpr_GInfo;     /* gadget context               */
    struct RastPort     *gpr_RPort;     /* all ready for use            */
    LONG                gpr_Redraw;     /* might be a "highlight pass"  */
};

/* values of gpr_Redraw */
#define GREDRAW_UPDATE  (2)     /* incremental update, e.g. prop slider */
#define GREDRAW_REDRAW  (1)     /* redraw gadget        */
#define GREDRAW_TOGGLE  (0)     /* toggle highlight, if applicable      */
```

```

/* GM_GOACTIVE, GM_HANDLEINPUT  */
struct gpInput {
    ULONG               MethodID;
    struct GadgetInfo   *gpi_GInfo;
    struct InputEvent   *gpi_IEvent;
    LONG                *gpi_Termination;
    struct {
        WORD    X;
        WORD    Y;
    }                   gpi_Mouse;
};

/* GM_HANDLEINPUT and GM_GOACTIVE  return code flags    */
/* return GMR_MEACTIVE (0) alone if you want more input.
 * Otherwise, return ONE of GMR_NOREUSE and GMR_REUSE, and optionally
 * GMR_VERIFY.
 */
#define GMR_MEACTIVE    (0)
#define GMR_NOREUSE     (1 << 1)
#define GMR_REUSE       (1 << 2)
#define GMR_VERIFY      (1 << 3)        /* you MUST set cgp_Termination */

/* New for V37:
 * You can end activation with one of GMR_NEXTACTIVE and GMR_PREVACTIVE,
 * which instructs Intuition to activate the next or previous gadget
 * that has GFLG_TABCYCLE set.
 */
#define GMR_NEXTACTIVE  (1 << 4)
#define GMR_PREVACTIVE  (1 << 5)

/* GM_GOINACTIVE */
struct gpGoInactive {
    ULONG               MethodID;
    struct GadgetInfo   *gpgi_GInfo;

    /* V37 field only!  DO NOT attempt to read under V36! */
    ULONG               gpgi_Abort;     /* gpgi_Abort=1 if gadget was aborted
                                         * by Intuition and 0 if gadget went
                                         * inactive at its own request
                                         */
};


/* Include obsolete identifiers: */
#ifndef INTUITION_IOBSOLETE_H
#include <intuition/iobsolete.h>
#endif

#endif
```

intuition/icclass.h

```
#ifndef INTUITION_ICCLASS_H
#define INTUITION_ICCLASS_H
/*
**  $Filename: intuition/icclass.h $
**  $Release: 2.04 $
**  $Revision: 36.1 $
**  $Date: 90/11/01 $
**
**  Gadget/object interconnection classes
**
**  (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/


#ifndef UTILITY_TAGITEM_H
#include <utility/tagitem.h>
#endif

#define ICM_Dummy       (0x0401L)       /* used for nothing              */
#define ICM_SETLOOP     (0x0402L)       /* set/increment loop counter    */
#define ICM_CLEARLOOP   (0x0403L)       /* clear/decrement loop counter */
#define ICM_CHECKLOOP   (0x0404L)       /* set/increment loop            */

/* no parameters for ICM_SETLOOP, ICM_CLEARLOOP, ICM_CHECKLOOP  */

/* interconnection attributes used by icclass, modelclass, and gadgetclass */
#define ICA_Dummy       (TAG_USER+0x40000L)
#define ICA_TARGET      (ICA_Dummy + 1)
        /* interconnection target               */
#define ICA_MAP         (ICA_Dummy + 2)
        /* interconnection map tagitem list     */
#define ICSPECIAL_CODE  (ICA_Dummy + 3)
        /* a "pseudo-attribute", see below.     */

/* Normally, the value for ICA_TARGET is some object pointer,
 * but if you specify the special value ICTARGET_IDCMP, notification
 * will be send as an IDCMP_IDCMPUPDATE message to the appropriate window's
 * IDCMP port.  See the definition of IDCMP_IDCMPUPDATE.
 *
 * When you specify ICTARGET_IDCMP for ICA_TARGET, the map you
 * specify will be applied to derive the attribute list that is
 * sent with the IDCMP_IDCMPUPDATE message.  If you specify a map list
 * which results in the attribute tag id ICSPECIAL_CODE, the
 * lower sixteen bits of the corresponding ti_Data value will
 * be copied into the Code field of the IDCMP_IDCMPUPDATE IntuiMessage.
 */
#define ICTARGET_IDCMP  (~0L)

#endif /* INTUITION_ICCLASS_H */
```

intuition/imageclass.h

```
#ifndef INTUITION_IMAGECLASS_H
#define INTUITION_IMAGECLASS_H TRUE
/*
**  $Filename: intuition/imageclass.h $
**  $Release: 2.04 $
**  $Revision: 36.1 $
**  $Date: 90/11/01 $
**
**  Definitions for the image classes
**
**  (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/


#ifndef UTILITY_TAGITEM_H
#include <utility/tagitem.h>
#endif

/*
 * NOTE:  <intuition/iobsolete.h> is included at the END of this file!
 */

#define CUSTOMIMAGEDEPTH        (-1)
/* if image.Depth is this, it's a new Image class object */

/* some convenient macros and casts */
#define GADGET_BOX( g )  ( (struct IBox *) &((struct Gadget *)(g))->LeftEdge )
#define IM_BOX( im )     ( (struct IBox *) &((struct Image *)(im))->LeftEdge )
#define IM_FGPEN( im )   ( (im)->PlanePick )
#define IM_BGPEN( im )   ( (im)->PlaneOnOff )

/***************************************************/
#define IA_Dummy                (TAG_USER + 0x20000)
#define IA_Left                 (IA_Dummy + 0x01)
#define IA_Top                  (IA_Dummy + 0x02)
#define IA_Width                (IA_Dummy + 0x03)
#define IA_Height               (IA_Dummy + 0x04)
#define IA_FGPen                (IA_Dummy + 0x05)
                    /* IA_FGPen also means "PlanePick"  */
#define IA_BGPen                (IA_Dummy + 0x06)
                    /* IA_BGPen also means "PlaneOnOff" */
#define IA_Data                 (IA_Dummy + 0x07)
                    /* bitplanes, for classic image,
                     * other image classes may use it for other things
                     */
#define IA_LineWidth            (IA_Dummy + 0x08)
#define IA_Pens                 (IA_Dummy + 0x0E)
                    /* pointer to UWORD pens[],
                     * ala DrawInfo.Pens, MUST be
                     * terminated by ~0.  Some classes can
                     * choose to have this, or SYSIA_DrawInfo,
                     * or both.
                     */
#define IA_Resolution           (IA_Dummy + 0x0F)
                    /* packed uwords for x/y resolution into a longword
                     * ala DrawInfo.Resolution
                     */

/**** see class documentation to learn which    *****/
/**** classes recognize these                   *****/
#define IA_APattern             (IA_Dummy + 0x10)
#define IA_APatSize             (IA_Dummy + 0x11)
#define IA_Mode                 (IA_Dummy + 0x12)
#define IA_Font                 (IA_Dummy + 0x13)
#define IA_Outline              (IA_Dummy + 0x14)
```

```
#define IA_Recessed                 (IA_Dummy + 0x15)
#define IA_DoubleEmboss             (IA_Dummy + 0x16)
#define IA_EdgesOnly                (IA_Dummy + 0x17)

/**** "sysiclass" attributes                          *****/
#define SYSIA_Size                  (IA_Dummy + 0x0B)
                        /* #define's below           */
#define SYSIA_Depth                 (IA_Dummy + 0x0C)
                        /* this is unused by Intuition.  SYSIA_DrawInfo
                         * is used instead for V36
                         */
#define SYSIA_Which                 (IA_Dummy + 0x0D)
                        /* see #define's below       */
#define SYSIA_DrawInfo              (IA_Dummy + 0x18)
                        /* pass to sysiclass, please */

/*****  obsolete: don't use these, use IA_Pens  *****/
#define SYSIA_Pens                  IA_Pens
#define IA_ShadowPen                (IA_Dummy + 0x09)
#define IA_HighlightPen             (IA_Dummy + 0x0A)

/** next attribute: (IA_Dummy + 0x19)   **/
/*************************************************/

/* data values for SYSIA_Size   */
#define SYSISIZE_MEDRES (0)
#define SYSISIZE_LOWRES (1)
#define SYSISIZE_HIRES  (2)

/*
 * SYSIA_Which tag data values:
 * Specifies which system gadget you want an image for.
 * Some numbers correspond to internal Intuition #defines
 */
#define DEPTHIMAGE      (0x00L)
#define ZOOMIMAGE       (0x01L)
#define SIZEIMAGE       (0x02L)
#define CLOSEIMAGE      (0x03L)
#define SDEPTHIMAGE     (0x05L) /* screen depth gadget */
#define LEFTIMAGE       (0x0AL)
#define UPIMAGE         (0x0BL)
#define RIGHTIMAGE      (0x0CL)
#define DOWNIMAGE       (0x0DL)
#define CHECKIMAGE      (0x0EL)
#define MXIMAGE         (0x0FL) /* mutual exclude "button" */


/* image message id's    */
#define    IM_DRAW      0x202L  /* draw yourself, with "state"          */
#define    IM_HITTEST   0x203L  /* return TRUE if click hits image      */
#define    IM_ERASE     0x204L  /* erase yourself                       */
#define    IM_MOVE      0x205L  /* draw new and erase old, smoothly     */

#define    IM_DRAWFRAME 0x206L  /* draw with specified dimensions       */
#define    IM_FRAMEBOX  0x207L  /* get recommended frame around some box*/
#define    IM_HITFRAME  0x208L  /* hittest with dimensions              */
#define    IM_ERASEFRAME 0x209L /* hittest with dimensions              */

/* image draw states or styles, for IM_DRAW */
#define    IDS_NORMAL           (0L)
#define    IDS_SELECTED         (1L)    /* for selected gadgets     */
#define    IDS_DISABLED         (2L)    /* for disabled gadgets     */
#define    IDS_BUSY             (3L)    /* for future functionality */
#define    IDS_INDETERMINATE    (4L)    /* for future functionality */
#define    IDS_INACTIVENORMAL   (5L)    /* normal, in inactive window border */
#define    IDS_INACTIVESELECTED (6L)    /* selected, in inactive border */
```

```
#define    IDS_INACTIVEDISABLED (7L)    /* disabled, in inactive border */

/* oops, please forgive spelling error by jimm */
#define IDS_INDETERMINANT IDS_INDETERMINATE

/* IM_FRAMEBOX  */
struct impFrameBox {
    ULONG               MethodID;
    struct IBox *imp_ContentsBox;       /* input: relative box of contents */
    struct IBox *imp_FrameBox;          /* output: rel. box of encl frame  */
    struct DrawInfo     *imp_DrInfo;
    ULONG       imp_FrameFlags;
};

#define FRAMEF_SPECIFY  (1<<0)  /* Make do with the dimensions of FrameBox
                                 * provided.
                                 */

/* IM_DRAW, IM_DRAWFRAME        */
struct impDraw {
    ULONG               MethodID;
    struct RastPort     *imp_RPort;
    struct {
        WORD    X;
        WORD    Y;
    }                   imp_Offset;

    ULONG               imp_State;
    struct DrawInfo     *imp_DrInfo;

    /* these parameters only valid for IM_DRAWFRAME */
    struct {
        WORD    Width;
        WORD    Height;
    }                   imp_Dimensions;
};

/* IM_ERASE, IM_ERASEFRAME      */
/* NOTE: This is a subset of impDraw    */
struct impErase {
    ULONG               MethodID;
    struct RastPort     *imp_RPort;
    struct {
        WORD    X;
        WORD    Y;
    }                   imp_Offset;

    /* these parameters only valid for IM_ERASEFRAME */
    struct {
        WORD    Width;
        WORD    Height;
    }                   imp_Dimensions;
};

/* IM_HITTEST, IM_HITFRAME      */
struct impHitTest {
    ULONG               MethodID;
    struct {
        WORD    X;
        WORD    Y;
    }                   imp_Point;

    /* these parameters only valid for IM_HITFRAME */
    struct {
        WORD    Width;
        WORD    Height;
```

intuition/imageclass.h Page 4

```
    }                       imp_Dimensions;
};

/* Include obsolete identifiers: */
#ifndef INTUITION_IOBSOLETE_H
#include <intuition/iobsolete.h>
#endif

#endif
```

intuition/intuition.h Page 1

```
#ifndef INTUITION_INTUITION_H
#define INTUITION_INTUITION_H TRUE
/*
**  $Filename: intuition/intuition.h $
**  $Release: 2.04 $
**  $Revision: 36.49 $
**  $Date: 91/02/22 $
**
**  Interface definitions for Intuition applications.
**
**  (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_TYPES_H
#include <exec/types.h>
#endif

#ifndef GRAPHICS_GFX_H
#include <graphics/gfx.h>
#endif

#ifndef GRAPHICS_CLIP_H
#include <graphics/clip.h>
#endif

#ifndef GRAPHICS_VIEW_H
#include <graphics/view.h>
#endif

#ifndef GRAPHICS_RASTPORT_H
#include <graphics/rastport.h>
#endif

#ifndef GRAPHICS_LAYERS_H
#include <graphics/layers.h>
#endif

#ifndef GRAPHICS_TEXT_H
#include <graphics/text.h>
#endif

#ifndef EXEC_PORTS_H
#include <exec/ports.h>
#endif

#ifndef DEVICES_INPUTEVENT_H
#include <devices/inputevent.h>
#endif

#ifndef UTILITY_TAGITEM_H
#include <utility/tagitem.h>
#endif

/*
 * NOTE:  intuition/iobsolete.h is included at the END of this file!
 */

/* ======================================================================== */
/* === Menu =============================================================== */
/* ======================================================================== */
struct Menu
{
    struct Menu *NextMenu;      /* same level */
    WORD LeftEdge, TopEdge;     /* position of the select box */
    WORD Width, Height; /* dimensions of the select box */
    UWORD Flags;                /* see flag definitions below */
    BYTE *MenuName;             /* text for this Menu Header */
    struct MenuItem *FirstItem; /* pointer to first in chain */

    /* these mysteriously-named variables are for internal use only */
    WORD JazzX, JazzY, BeatX, BeatY;
};


/* FLAGS SET BY BOTH THE APPLIPROG AND INTUITION */
#define MENUENABLED 0x0001      /* whether or not this menu is enabled */

/* FLAGS SET BY INTUITION */
#define MIDRAWN 0x0100          /* this menu's items are currently drawn */






/* ======================================================================== */
/* === MenuItem =========================================================== */
/* ======================================================================== */
struct MenuItem
{
    struct MenuItem *NextItem;  /* pointer to next in chained list */
    WORD LeftEdge, TopEdge;     /* position of the select box */
    WORD Width, Height;         /* dimensions of the select box */
    UWORD Flags;                /* see the defines below */

    LONG MutualExclude;         /* set bits mean this item excludes that */

    APTR ItemFill;              /* points to Image, IntuiText, or NULL */

    /* when this item is pointed to by the cursor and the items highlight
     *  mode HIGHIMAGE is selected, this alternate image will be displayed
     */
    APTR SelectFill;            /* points to Image, IntuiText, or NULL */

    BYTE Command;               /* only if appliprog sets the COMMSEQ flag */

    struct MenuItem *SubItem;   /* if non-zero, points to MenuItem for submenu *
  /

    /* The NextSelect field represents the menu number of next selected
     *  item (when user has drag-selected several items)
     */
    UWORD NextSelect;
};


/* FLAGS SET BY THE APPLIPROG */
#define CHECKIT         0x0001  /* set to indicate checkmarkable item */
#define ITEMTEXT        0x0002  /* set if textual, clear if graphical item */
#define COMMSEQ         0x0004  /* set if there's an command sequence */
#define MENUTOGGLE      0x0008  /* set for toggling checks (else mut. exclude) *
  /
#define ITEMENABLED     0x0010  /* set if this item is enabled */

/* these are the SPECIAL HIGHLIGHT FLAG state meanings */
#define HIGHFLAGS       0x00C0  /* see definitions below for these bits */
#define HIGHIMAGE       0x0000  /* use the user's "select image" */
#define HIGHCOMP        0x0040  /* highlight by complementing the selectbox */
#define HIGHBOX         0x0080  /* highlight by "boxing" the selectbox */
#define HIGHNONE        0x00C0  /* don't highlight */

```

```
/* FLAGS SET BY BOTH APPLIPROG AND INTUITION */
#define CHECKED 0x0100  /* state of the checkmark */

/* FLAGS SET BY INTUITION */
#define ISDRAWN         0x1000  /* this item's subs are currently drawn */
#define HIGHITEM        0x2000  /* this item is currently highlighted */
#define MENUTOGGLED     0x4000  /* this item was already toggled */





/* ======================================================================== */
/* === Requester ========================================================== */
/* ======================================================================== */
struct Requester
{
    struct Requester *OlderRequest;
    WORD LeftEdge, TopEdge;             /* dimensions of the entire box */
    WORD Width, Height;                 /* dimensions of the entire box */
    WORD RelLeft, RelTop;               /* for Pointer relativity offsets */

    struct Gadget *ReqGadget;           /* pointer to a list of Gadgets */
    struct Border *ReqBorder;           /* the box's border */
    struct IntuiText *ReqText;          /* the box's text */
    UWORD Flags;                        /* see definitions below */

    /* pen number for back-plane fill before draws */
    UBYTE BackFill;
    /* Layer in place of clip rect      */
    struct Layer *ReqLayer;

    UBYTE ReqPad1[32];

    /* If the BitMap plane pointers are non-zero, this tells the system
     * that the image comes pre-drawn (if the appliprog wants to define
     * its own box, in any shape or size it wants!);  this is OK by
     * Intuition as long as there's a good correspondence between
     * the image and the specified Gadgets
     */
    struct BitMap *ImageBMap;   /* points to the BitMap of PREDRAWN imagery */
    struct Window *RWindow;     /* added.  points back to Window */

    struct Image  *ReqImage;    /* new for V36: drawn if USEREQIMAGE set */

    UBYTE ReqPad2[32];
};


/* FLAGS SET BY THE APPLIPROG */
#define POINTREL        0x0001
                          /* if POINTREL set, TopLeft is relative to pointer
                           * for DMRequester, relative to window center
                           * for Request().
                           */
#define PREDRAWN        0x0002
        /* set if Requester.ImageBMap points to predrawn Requester imagery */
#define NOISYREQ        0x0004
        /* if you don't want requester to filter input     */
#define SIMPLEREQ       0x0010
        /* to use SIMPLEREFRESH layer (recommended)     */

/* New for V36          */
#define USEREQIMAGE     0x0020
        /*  render linked list ReqImage after BackFill
         * but before gadgets and text
        */
#define NOREQBACKFILL   0x0040
        /* don't bother filling requester with Requester.BackFill pen   */


/* FLAGS SET BY INTUITION */
#define REQOFFWINDOW    0x1000  /* part of one of the Gadgets was offwindow */
#define REQACTIVE       0x2000  /* this requester is active */
#define SYSREQUEST      0x4000  /* this requester caused by system */
#define DEFERREFRESH    0x8000  /* this Requester stops a Refresh broadcast */






/* ====================================================================== */
/* === Gadget =========================================================== */
/* ====================================================================== */
struct Gadget
{
    struct Gadget *NextGadget;  /* next gadget in the list */

    WORD LeftEdge, TopEdge;     /* "hit box" of gadget */
    WORD Width, Height;         /* "hit box" of gadget */

    UWORD Flags;                /* see below for list of defines */

    UWORD Activation;           /* see below for list of defines */

    UWORD GadgetType;           /* see below for defines */

    /* appliprog can specify that the Gadget be rendered as either as Border
     * or an Image.  This variable points to which (or equals NULL if there's
     * nothing to be rendered about this Gadget)
     */
    APTR GadgetRender;

    /* appliprog can specify "highlighted" imagery rather than algorithmic
     * this can point to either Border or Image data
     */
    APTR SelectRender;

    struct IntuiText *GadgetText;   /* text for this gadget */

    /* MutualExclude, never implemented, is now declared obsolete.
     * There are published examples of implementing a more general
     * and practical exclusion in your applications.
     *
     * Starting with V36, this field is used to point to a hook
     * for a custom gadget.
     *
     * Programs using this field for their own processing will
     * continue to work, as long as they don't try the
     * trick with custom gadgets.
     */
    LONG MutualExclude;  /* obsolete */

    /* pointer to a structure of special data required by Proportional,
     * String and Integer Gadgets
     */
    APTR SpecialInfo;

    UWORD GadgetID;     /* user-definable ID field */
    APTR UserData;      /* ptr to general purpose User data (ignored by In) */
};
```

```


/* --- Gadget.Flags values      --- */
/* combinations in these bits describe the highlight technique to be used */
#define GFLG_GADGHIGHBITS 0x0003
#define GFLG_GADGHCOMP    0x0000  /* Complement the select box */
#define GFLG_GADGHBOX     0x0001  /* Draw a box around the image */
#define GFLG_GADGHIMAGE   0x0002  /* Blast in this alternate image */
#define GFLG_GADGHNONE    0x0003  /* don't highlight */

#define GFLG_GADGIMAGE            0x0004  /* set if GadgetRender and SelectRende
  r
                                           * point to an Image structure, clear
                                           * if they point to Border structures
                                           */

/* combinations in these next two bits specify to which corner the gadget's
 *  Left & Top coordinates are relative.  If relative to Top/Left,
 *  these are "normal" coordinates (everything is relative to something in
 *  this universe).
 *
 * Gadget positions and dimensions are relative to the window or
 * requester which contains the gadget
 */
#define GFLG_RELBOTTOM    0x0008  /* vert. pos. is relative to bottom edge */
#define GFLG_RELRIGHT     0x0010  /* horiz. pos. is relative to right edge */
#define GFLG_RELWIDTH     0x0020  /* width is relative to req/window    */
#define GFLG_RELHEIGHT    0x0040  /* height is relative to req/window   */

#define GFLG_SELECTED     0x0080  /* you may initialize and look at this
  */

/* the GFLG_DISABLED flag is initialized by you and later set by Intuition
 * according to your calls to On/OffGadget().  It specifies whether or not
 * this Gadget is currently disabled from being selected
 */
#define GFLG_DISABLED     0x0100

/* These flags specify the type of text field that Gadget.GadgetText
 * points to.  In all normal (pre-V36) gadgets which you initialize
 * this field should always be zero.  Some types of gadget objects
 * created from classes will use these fields to keep track of
 * types of labels/contents that different from IntuiText, but are
 * stashed in GadgetText.
 */

#define GFLG_LABELMASK    0x3000
#define GFLG_LABELITEXT   0x0000  /* GadgetText points to IntuiText     */
#define GFLG_LABELSTRING  0x1000  /* GadgetText points to (UBYTE *)     */
#define GFLG_LABELIMAGE   0x2000  /* GadgetText points to Image (object)
  */

/* New for V37: GFLG_TABCYCLE */
#define GFLG_TABCYCLE     0x0200  /* (string or custom) gadget participates in
                                   * cycling activation with Tab or Shift-Tab
                                   */
/* New for V37: GFLG_STRINGEXTEND.  We discovered that V34 doesn't properly
 * ignore the value we had chosen for the Gadget->Activation flag
 * GACT_STRINGEXTEND.  NEVER SET THAT FLAG WHEN RUNNING UNDER V34.
 * The Gadget->Flags bit GFLG_STRINGEXTEND is provided as a synonym which is
 * safe under V34, and equivalent to GACT_STRINGEXTEND under V37.
 * (Note that the two flags are not numerically equal)
 */
#define GFLG_STRINGEXTEND 0x0400  /* this String Gadget has StringExtend
  */

```

```
/* ---  Gadget.Activation flag values   --- */
/* Set GACT_RELVERIFY if you want to verify that the pointer was still over
 * the gadget when the select button was released.  Will cause
 * an IDCMP_GADGETUP message to be sent if so.
 */
#define GACT_RELVERIFY    0x0001

/* the flag GACT_IMMEDIATE, when set, informs the caller that the gadget
 *  was activated when it was activated.  This flag works in conjunction with
 *  the GACT_RELVERIFY flag
 */
#define GACT_IMMEDIATE    0x0002

/* the flag GACT_ENDGADGET, when set, tells the system that this gadget,
 * when selected, causes the Requester to be ended.  Requesters
 * that are ended are erased and unlinked from the system.
 */
#define GACT_ENDGADGET    0x0004

/* the GACT_FOLLOWMOUSE flag, when set, specifies that you want to receive
 * reports on mouse movements while this gadget is active.
 * You probably want to set the GACT_IMMEDIATE flag when using
 * GACT_FOLLOWMOUSE, since that's the only reasonable way you have of
 * learning why Intuition is suddenly sending you a stream of mouse
 * movement events.  If you don't set GACT_RELVERIFY, you'll get at
 * least one Mouse Position event.
 */
#define GACT_FOLLOWMOUSE  0x0008

/* if any of the BORDER flags are set in a Gadget that's included in the
 * Gadget list when a Window is opened, the corresponding Border will
 * be adjusted to make room for the Gadget
 */
#define GACT_RIGHTBORDER  0x0010
#define GACT_LEFTBORDER   0x0020
#define GACT_TOPBORDER    0x0040
#define GACT_BOTTOMBORDER 0x0080
#define GACT_BORDERSNIFF  0x8000  /* neither set nor rely on this bit   */

#define GACT_TOGGLESELECT 0x0100  /* this bit for toggle-select mode */
#define GACT_BOOLEXTEND   0x2000  /* this Boolean Gadget has a BoolInfo */

/* should properly be in StringInfo, but aren't */
#define GACT_STRINGLEFT   0x0000  /* NOTE WELL: that this has value zero
   */
#define GACT_STRINGCENTER 0x0200
#define GACT_STRINGRIGHT  0x0400
#define GACT_LONGINT      0x0800  /* this String Gadget is for Long Ints
   */
#define GACT_ALTKEYMAP    0x1000  /* this String has an alternate keymap
   */
#define GACT_STRINGEXTEND 0x2000  /* this String Gadget has StringExtend
   */
                                  /* NOTE: NEVER SET GACT_STRINGEXTEND IF YOU
                                   * ARE RUNNING ON LESS THAN V36!  SEE
                                   * GFLG_STRINGEXTEND (ABOVE) INSTEAD
                                   */

#define GACT_ACTIVEGADGET 0x4000  /* this gadget is "active".  This flag
                                   * is maintained by Intuition, and you
                                   * cannot count on its value persisting
                                   * while you do something on your program's
                                   * task.  It can only be trusted by
                                   * people implementing custom gadgets
                                   */

```

```
/* note 0x8000 is used above (GACT_BORDERSNIFF);
 * all Activation flags defined */

/* --- GADGET TYPES ------------------------------------------------------- */
/* These are the Gadget Type definitions for the variable GadgetType
 * gadget number type MUST start from one.  NO TYPES OF ZERO ALLOWED.
 * first comes the mask for Gadget flags reserved for Gadget typing
 */
#define GTYP_GADGETTYPE 0xFC00  /* all Gadget Global Type flags (padded) */
#define GTYP_SYSGADGET  0x8000  /* 1 = Allocated by the system, 0 = by app. */
#define GTYP_SCRGADGET  0x4000  /* 1 = ScreenGadget, 0 = WindowGadget */
#define GTYP_GZZGADGET  0x2000  /* 1 = for WFLG_GIMMEZEROZERO borders */
#define GTYP_REQGADGET  0x1000  /* 1 = this is a Requester Gadget */
/* system gadgets */
#define GTYP_SIZING     0x0010
#define GTYP_WDRAGGING  0x0020
#define GTYP_SDRAGGING  0x0030
#define GTYP_WUPFRONT   0x0040
#define GTYP_SUPFRONT   0x0050
#define GTYP_WDOWNBACK  0x0060
#define GTYP_SDOWNBACK  0x0070
#define GTYP_CLOSE      0x0080
/* application gadgets */
#define GTYP_BOOLGADGET 0x0001
#define GTYP_GADGET0002 0x0002
#define GTYP_PROPGADGET 0x0003
#define GTYP_STRGADGET  0x0004
#define GTYP_CUSTOMGADGET       0x0005
#define GTYP_GTYPEMASK  0x0007  /* masks out to gadget class    */

/* This bit in GadgetType is reserved for undocumented internal use
 * by the Gadget Toolkit, and cannot be used nor relied on by
 * applications:        0x0100
 */

/* ======================================================================= */
/* === BoolInfo======================================================= */
/* ======================================================================= */
/* This is the special data needed by an Extended Boolean Gadget
 * Typically this structure will be pointed to by the Gadget field SpecialInfo
 */
struct BoolInfo
{
    UWORD  Flags;       /* defined below */
    UWORD  *Mask;       /* bit mask for highlighting and selecting
                         * mask must follow the same rules as an Image
                         * plane.  Its width and height are determined
                         * by the width and height of the gadget's
                         * select box. (i.e. Gadget.Width and .Height).
                         */
    ULONG  Reserved;    /* set to 0     */
};

/* set BoolInfo.Flags to this flag bit.
 * in the future, additional bits might mean more stuff hanging
 * off of BoolInfo.Reserved.
 */
#define BOOLMASK        0x0001  /* extension is for masked gadget */

/* ======================================================================= */
/* === PropInfo =========================================================== */
/* ======================================================================= */
/* this is the special data required by the proportional Gadget
 * typically, this data will be pointed to by the Gadget variable SpecialInfo
 */
struct PropInfo
```

```
{
    UWORD Flags;        /* general purpose flag bits (see defines below) */

    /* You initialize the Pot variables before the Gadget is added to
     * the system.  Then you can look here for the current settings
     * any time, even while User is playing with this Gadget.  To
     * adjust these after the Gadget is added to the System, use
     * ModifyProp();  The Pots are the actual proportional settings,
     * where a value of zero means zero and a value of MAXPOT means
     * that the Gadget is set to its maximum setting.
     */
    UWORD HorizPot;     /* 16-bit FixedPoint horizontal quantity percentage */
    UWORD VertPot;      /* 16-bit FixedPoint vertical quantity percentage */

    /* the 16-bit FixedPoint Body variables describe what percentage of
     * the entire body of stuff referred to by this Gadget is actually
     * shown at one time.  This is used with the AUTOKNOB routines,
     * to adjust the size of the AUTOKNOB according to how much of
     * the data can be seen.  This is also used to decide how far
     * to advance the Pots when User hits the Container of the Gadget.
     * For instance, if you were controlling the display of a 5-line
     * Window of text with this Gadget, and there was a total of 15
     * lines that could be displayed, you would set the VertBody value to
     *     (MAXBODY / (TotalLines / DisplayLines)) = MAXBODY / 3.
     * Therefore, the AUTOKNOB would fill 1/3 of the container, and
     * if User hits the Cotainer outside of the knob, the pot would
     * advance 1/3 (plus or minus) If there's no body to show, or
     * the total amount of displayable info is less than the display area,
     * set the Body variables to the MAX.  To adjust these after the
     * Gadget is added to the System, use ModifyProp();
     */
    UWORD HorizBody;            /* horizontal Body */
    UWORD VertBody;             /* vertical Body */

    /* these are the variables that Intuition sets and maintains */
    UWORD CWidth;       /* Container width (with any relativity absoluted) */
    UWORD CHeight;      /* Container height (with any relativity absoluted) */
    UWORD HPotRes, VPotRes;     /* pot increments */
    UWORD LeftBorder;           /* Container borders */
    UWORD TopBorder;            /* Container borders */
};


/* --- FLAG BITS ---------------------------------------------------------- */
#define AUTOKNOB        0x0001  /* this flag sez:  gimme that old auto-knob */
/* NOTE: if you do not use an AUTOKNOB for a proportional gadget,
 * you are currently limited to using a single Image of your own
 * design: Intuition won't handle a linked list of images as
 * a proportional gadget knob.
 */

#define FREEHORIZ       0x0002  /* if set, the knob can move horizontally */
#define FREEVERT        0x0004  /* if set, the knob can move vertically */
#define PROPBORDERLESS  0x0008  /* if set, no border will be rendered */
#define KNOBHIT         0x0100  /* set when this Knob is hit */
#define PROPNEWLOOK     0x0010  /* set this if you want to get the new
                                 * V36 look
                                 */

#define KNOBHMIN        6       /* minimum horizontal size of the Knob */
#define KNOBVMIN        4       /* minimum vertical size of the Knob */
#define MAXBODY         0xFFFF  /* maximum body value */
#define MAXPOT                  0xFFFF  /* maximum pot value */


/* ======================================================================== */
```

```
/* === StringInfo ========================================================= */
/* ======================================================================== */
/* this is the special data required by the string Gadget
 * typically, this data will be pointed to by the Gadget variable SpecialInfo
 */
struct StringInfo
{
    /* you initialize these variables, and then Intuition maintains them */
    UBYTE *Buffer;      /* the buffer containing the start and final string */
    UBYTE *UndoBuffer;  /* optional buffer for undoing current entry */
    WORD BufferPos;     /* character position in Buffer */
    WORD MaxChars;      /* max number of chars in Buffer (including NULL) */
    WORD DispPos;       /* Buffer position of first displayed character */

    /* Intuition initializes and maintains these variables for you */
    WORD UndoPos;       /* character position in the undo buffer */
    WORD NumChars;      /* number of characters currently in Buffer */
    WORD DispCount;     /* number of whole characters visible in Container */
    WORD CLeft, CTop;   /* topleft offset of the container */

    /* This unused field is changed to allow extended specification
     * of string gadget parameters.  It is ignored unless the flag
     * GACT_STRINGEXTEND is set in the Gadget's Activation field
     * or the GFLG_STRINGEXTEND flag is set in the Gadget Flags field.
     * (See GFLG_STRINGEXTEND for an important note)
     */
    /* struct Layer *LayerPtr;  --- obsolete --- */
    struct StringExtend *Extension;

    /* you can initialize this variable before the gadget is submitted to
     * Intuition, and then examine it later to discover what integer
     * the user has entered (if the user never plays with the gadget,
     * the value will be unchanged from your initial setting)
     */
    LONG LongInt;

    /* If you want this Gadget to use your own Console keymapping, you
     * set the GACT_ALTKEYMAP bit in the Activation flags of the Gadget,
     * and then set this variable to point to your keymap.  If you don't
     * set the GACT_ALTKEYMAP, you'll get the standard ASCII keymapping.
     */
    struct KeyMap *AltKeyMap;
};

/* ======================================================================== */
/* === IntuiText ========================================================== */
/* ======================================================================== */
/* IntuiText is a series of strings that start with a location
 *  (always relative to the upper-left corner of something) and then the
 *  text of the string.  The text is null-terminated.
 */
struct IntuiText
{
    UBYTE FrontPen, BackPen;    /* the pen numbers for the rendering */
    UBYTE DrawMode;             /* the mode for rendering the text */
    WORD LeftEdge;              /* relative start location for the text */
    WORD TopEdge;               /* relative start location for the text */
    struct TextAttr *ITextFont; /* if NULL, you accept the default */
    UBYTE *IText;               /* pointer to null-terminated text */
    struct IntuiText *NextText; /* pointer to another IntuiText to render */
};
```

```
/* ========================================================================= */
/* === Border ============================================================== */
/* ========================================================================= */
/* Data type Border, used for drawing a series of lines which is intended for
 *  use as a border drawing, but which may, in fact, be used to render any
 *  arbitrary vector shape.
 *  The routine DrawBorder sets up the RastPort with the appropriate
 *  variables, then does a Move to the first coordinate, then does Draws
 *  to the subsequent coordinates.
 *  After all the Draws are done, if NextBorder is non-zero we call DrawBorder
 *  on NextBorder
 */
struct Border
{
    WORD LeftEdge, TopEdge;     /* initial offsets from the origin */
    UBYTE FrontPen, BackPen;    /* pens numbers for rendering */
    UBYTE DrawMode;             /* mode for rendering */
    BYTE Count;                 /* number of XY pairs */
    WORD *XY;                   /* vector coordinate pairs rel to LeftTop */
    struct Border *NextBorder;  /* pointer to any other Border too */
};






/* ========================================================================= */
/* === Image =============================================================== */
/* ========================================================================= */
/* This is a brief image structure for very simple transfers of
 * image data to a RastPort
 */
struct Image
{
    WORD LeftEdge;              /* starting offset relative to some origin */
    WORD TopEdge;               /* starting offsets relative to some origin */
    WORD Width;                 /* pixel size (though data is word-aligned) */
    WORD Height;
    WORD Depth;                 /* >= 0, for images you create         */
    UWORD *ImageData;           /* pointer to the actual word-aligned bits */

    /* the PlanePick and PlaneOnOff variables work much the same way as the
     * equivalent GELS Bob variables.  It's a space-saving
     * mechanism for image data.  Rather than defining the image data
     * for every plane of the RastPort, you need define data only
     * for the planes that are not entirely zero or one.  As you
     * define your Imagery, you will often find that most of the planes
     * ARE just as color selectors.  For instance, if you're designing
     * a two-color Gadget to use colors one and three, and the Gadget
     * will reside in a five-plane display, bit plane zero of your
     * imagery would be all ones, bit plane one would have data that
     * describes the imagery, and bit planes two through four would be
     * all zeroes.  Using these flags avoids wasting all
     * that memory in this way:  first, you specify which planes you
     * want your data to appear in using the PlanePick variable.  For
     * each bit set in the variable, the next "plane" of your image
     * data is blitted to the display.  For each bit clear in this
     * variable, the corresponding bit in PlaneOnOff is examined.
     * If that bit is clear, a "plane" of zeroes will be used.
     * If the bit is set, ones will go out instead.  So, for our example:
     *   Gadget.PlanePick = 0x02;
     *   Gadget.PlaneOnOff = 0x01;
     * Note that this also allows for generic Gadgets, like the
     * System Gadgets, which will work in any number of bit planes.
```

```
     * Note also that if you want an Image that is only a filled
     * rectangle, you can get this by setting PlanePick to zero
     * (pick no planes of data) and set PlaneOnOff to describe the pen
     * color of the rectangle.
     *
     * NOTE:  Intuition relies on PlanePick to know how many planes
     * of data are found in ImageData.  There should be no more
     * '1'-bits in PlanePick than there are planes in ImageData.
     */
    UBYTE PlanePick, PlaneOnOff;

    /* if the NextImage variable is not NULL, Intuition presumes that
     * it points to another Image structure with another Image to be
     * rendered
     */
    struct Image *NextImage;
};






/* ========================================================================= */
/* === IntuiMessage ======================================================== */
/* ========================================================================= */
struct IntuiMessage
{
    struct Message ExecMessage;

    /* the Class bits correspond directly with the IDCMP Flags, except for the
     * special bit IDCMP_LONELYMESSAGE (defined below)
     */
    ULONG Class;

    /* the Code field is for special values like MENU number */
    UWORD Code;

    /* the Qualifier field is a copy of the current InputEvent's Qualifier */
    UWORD Qualifier;

    /* IAddress contains particular addresses for Intuition functions, like
     * the pointer to the Gadget or the Screen
     */
    APTR IAddress;

    /* when getting mouse movement reports, any event you get will have the
     * the mouse coordinates in these variables.  the coordinates are relative
     * to the upper-left corner of your Window (WFLG_GIMMEZEROZERO
     * notwithstanding).  If IDCMP_DELTAMOVE is set, these values will
     * be deltas from the last reported position.
     */
    WORD MouseX, MouseY;

    /* the time values are copies of the current system clock time.  Micros
     * are in units of microseconds, Seconds in seconds.
     */
    ULONG Seconds, Micros;

    /* the IDCMPWindow variable will always have the address of the Window of
     * this IDCMP
     */
    struct Window *IDCMPWindow;

    /* system-use variable */
    struct IntuiMessage *SpecialLink;
```

```
};


/* --- IDCMP Classes -------------------------------------------------------- */
/* Please refer to the Autodoc for OpenWindow() and to the Rom Kernel
 * Manual for full details on the IDCMP classes.
 */
#define IDCMP_SIZEVERIFY        0x00000001
#define IDCMP_NEWSIZE           0x00000002
#define IDCMP_REFRESHWINDOW     0x00000004
#define IDCMP_MOUSEBUTTONS      0x00000008
#define IDCMP_MOUSEMOVE         0x00000010
#define IDCMP_GADGETDOWN        0x00000020
#define IDCMP_GADGETUP          0x00000040
#define IDCMP_REQSET            0x00000080
#define IDCMP_MENUPICK          0x00000100
#define IDCMP_CLOSEWINDOW       0x00000200
#define IDCMP_RAWKEY            0x00000400
#define IDCMP_REQVERIFY         0x00000800
#define IDCMP_REQCLEAR          0x00001000
#define IDCMP_MENUVERIFY        0x00002000
#define IDCMP_NEWPREFS          0x00004000
#define IDCMP_DISKINSERTED      0x00008000
#define IDCMP_DISKREMOVED       0x00010000
#define IDCMP_WBENCHMESSAGE     0x00020000  /*  System use only         */
#define IDCMP_ACTIVEWINDOW      0x00040000
#define IDCMP_INACTIVEWINDOW    0x00080000
#define IDCMP_DELTAMOVE         0x00100000
#define IDCMP_VANILLAKEY        0x00200000
#define IDCMP_INTUITICKS        0x00400000
/*  for notifications from "boopsi" gadgets     */
#define IDCMP_IDCMPUPDATE       0x00800000  /* new for V36      */
/* for getting help key report during menu session      */
#define IDCMP_MENUHELP          0x01000000  /* new for V36      */
/* for notification of any move/size/zoom/change window     */
#define IDCMP_CHANGEWINDOW      0x02000000  /* new for V36      */

/* NOTEZ-BIEN:                          0x80000000 is reserved for internal use
    */

/* the IDCMP Flags do not use this special bit, which is cleared when
 * Intuition sends its special message to the Task, and set when Intuition
 * gets its Message back from the Task.  Therefore, I can check here to
 * find out fast whether or not this Message is available for me to send
 */
#define IDCMP_LONELYMESSAGE     0x80000000


/* --- IDCMP Codes ---------------------------------------------------------- */
/* This group of codes is for the IDCMP_MENUVERIFY function */
#define MENUHOT         0x0001  /* IntuiWants verification or MENUCANCEL    */
#define MENUCANCEL      0x0002  /* HOT Reply of this cancels Menu operation */
#define MENUWAITING     0x0003  /* Intuition simply wants a ReplyMsg() ASAP */

/* These are internal tokens to represent state of verification attempts
 * shown here as a clue.
 */
#define OKOK            MENUHOT /* guy didn't care                      */
#define OKABORT         0x0004  /* window rendered question moot        */
#define OKCANCEL        MENUCANCEL /* window sent cancel reply          */

/* This group of codes is for the IDCMP_WBENCHMESSAGE messages */
#define WBENCHOPEN      0x0001
#define WBENCHCLOSE     0x0002


```

```
/* A data structure common in V36 Intuition processing  */
struct IBox {
    WORD Left;
    WORD Top;
    WORD Width;
    WORD Height;
    };



/* ========================================================================== */
/* === Window =============================================================== */
/* ========================================================================== */
struct Window
{
    struct Window *NextWindow;          /* for the linked list in a screen */

    WORD LeftEdge, TopEdge;             /* screen dimensions of window */
    WORD Width, Height;                 /* screen dimensions of window */

    WORD MouseY, MouseX;                /* relative to upper-left of window */

    WORD MinWidth, MinHeight;           /* minimum sizes */
    UWORD MaxWidth, MaxHeight;          /* maximum sizes */

    ULONG Flags;                        /* see below for defines */

    struct Menu *MenuStrip;             /* the strip of Menu headers */

    UBYTE *Title;                       /* the title text for this window */

    struct Requester *FirstRequest;     /* all active Requesters */

    struct Requester *DMRequest;        /* double-click Requester */

    WORD ReqCount;                      /* count of reqs blocking Window */

    struct Screen *WScreen;             /* this Window's Screen */
    struct RastPort *RPort;             /* this Window's very own RastPort */

    /* the border variables describe the window border.  If you specify
     * WFLG_GIMMEZEROZERO when you open the window, then the upper-left of
     * the ClipRect for this window will be upper-left of the BitMap (with
     * correct offsets when in SuperBitMap mode; you MUST select
     * WFLG_GIMMEZEROZERO when using SuperBitMap).  If you don't specify
     * ZeroZero, then you save memory (no allocation of RastPort, Layer,
     * ClipRect and associated Bitmaps), but you also must offset all your
     * writes by BorderTop, BorderLeft and do your own mini-clipping to
     * prevent writing over the system gadgets
     */
    BYTE BorderLeft, BorderTop, BorderRight, BorderBottom;
    struct RastPort *BorderRPort;


    /* You supply a linked-list of Gadgets for your Window.
     * This list DOES NOT include system gadgets.  You get the standard
     * window system gadgets by setting flag-bits in the variable Flags (see
     * the bit definitions below)
     */
    struct Gadget *FirstGadget;

    /* these are for opening/closing the windows */
    struct Window *Parent, *Descendant;

    /* sprite data information for your own Pointer
     * set these AFTER you Open the Window by calling SetPointer()
```

```

/* ======================================================================== */
/* === Border ============================================================= */
/* ======================================================================== */
/* Data type Border, used for drawing a series of lines which is intended for
 *  use as a border drawing, but which may, in fact, be used to render any
 *  arbitrary vector shape.
 *  The routine DrawBorder sets up the RastPort with the appropriate
 *  variables, then does a Move to the first coordinate, then does Draws
 *  to the subsequent coordinates.
 *  After all the Draws are done, if NextBorder is non-zero we call DrawBorder
 *  on NextBorder
 */
struct Border
{
    WORD LeftEdge, TopEdge;     /* initial offsets from the origin */
    UBYTE FrontPen, BackPen;    /* pens numbers for rendering */
    UBYTE DrawMode;             /* mode for rendering */
    BYTE Count;                 /* number of XY pairs */
    WORD *XY;                   /* vector coordinate pairs rel to LeftTop */
    struct Border *NextBorder;  /* pointer to any other Border too */
};






/* ======================================================================== */
/* === Image ============================================================== */
/* ======================================================================== */
/* This is a brief image structure for very simple transfers of
 * image data to a RastPort
 */
struct Image
{
    WORD LeftEdge;              /* starting offset relative to some origin */
    WORD TopEdge;               /* starting offsets relative to some origin */
    WORD Width;                 /* pixel size (though data is word-aligned) */
    WORD Height;
    WORD Depth;                 /* >= 0, for images you create         */
    UWORD *ImageData;           /* pointer to the actual word-aligned bits */

    /* the PlanePick and PlaneOnOff variables work much the same way as the
     * equivalent GELS Bob variables.  It's a space-saving
     * mechanism for image data.  Rather than defining the image data
     * for every plane of the RastPort, you need define data only
     * for the planes that are not entirely zero or one.  As you
     * define your Imagery, you will often find that most of the planes
     * ARE just as color selectors.  For instance, if you're designing
     * a two-color Gadget to use colors one and three, and the Gadget
     * will reside in a five-plane display, bit plane zero of your
     * imagery would be all ones, bit plane one would have data that
     * describes the imagery, and bit planes two through four would be
     * all zeroes.  Using these flags avoids wasting all
     * that memory in this way:  first, you specify which planes you
     * want your data to appear in using the PlanePick variable.  For
     * each bit set in the variable, the next "plane" of your image
     * data is blitted to the display.  For each bit clear in this
     * variable, the corresponding bit in PlaneOnOff is examined.
     * If that bit is clear, a "plane" of zeroes will be used.
     * If the bit is set, ones will go out instead.  So, for our example:
     *   Gadget.PlanePick = 0x02;
     *   Gadget.PlaneOnOff = 0x01;
     * Note that this also allows for generic Gadgets, like the
     * System Gadgets, which will work in any number of bit planes.
```

```
     * Note also that if you want an Image that is only a filled
     * rectangle, you can get this by setting PlanePick to zero
     * (pick no planes of data) and set PlaneOnOff to describe the pen
     * color of the rectangle.
     *
     * NOTE:  Intuition relies on PlanePick to know how many planes
     * of data are found in ImageData.  There should be no more
     * '1'-bits in PlanePick than there are planes in ImageData.
     */
    UBYTE PlanePick, PlaneOnOff;

    /* if the NextImage variable is not NULL, Intuition presumes that
     * it points to another Image structure with another Image to be
     * rendered
     */
    struct Image *NextImage;
};






/* ======================================================================== */
/* === IntuiMessage ======================================================= */
/* ======================================================================== */
struct IntuiMessage
{
    struct Message ExecMessage;

    /* the Class bits correspond directly with the IDCMP Flags, except for the
     * special bit IDCMP_LONELYMESSAGE (defined below)
     */
    ULONG Class;

    /* the Code field is for special values like MENU number */
    UWORD Code;

    /* the Qualifier field is a copy of the current InputEvent's Qualifier */
    UWORD Qualifier;

    /* IAddress contains particular addresses for Intuition functions, like
     * the pointer to the Gadget or the Screen
     */
    APTR IAddress;

    /* when getting mouse movement reports, any event you get will have the
     * the mouse coordinates in these variables.  the coordinates are relative
     * to the upper-left corner of your Window (WFLG_GIMMEZEROZERO
     * notwithstanding).  If IDCMP_DELTAMOVE is set, these values will
     * be deltas from the last reported position.
     */
    WORD MouseX, MouseY;

    /* the time values are copies of the current system clock time.  Micros
     * are in units of microseconds, Seconds in seconds.
     */
    ULONG Seconds, Micros;

    /* the IDCMPWindow variable will always have the address of the Window of
     * this IDCMP
     */
    struct Window *IDCMPWindow;

    /* system-use variable */
    struct IntuiMessage *SpecialLink;

    ULONG IDCMPFlags;               /* User-selected IDCMP flags */

    ULONG Flags;                    /* see Window struct for defines */

    /* You supply a linked-list of Gadgets for your Window.
     *  This list DOES NOT include system Gadgets.  You get the standard
     *  system Window Gadgets by setting flag-bits in the variable Flags (see
     *  the bit definitions under the Window structure definition)
     */
    struct Gadget *FirstGadget;

    /* the CheckMark is a pointer to the imagery that will be used when
     * rendering MenuItems of this Window that want to be checkmarked
     * if this is equal to NULL, you'll get the default imagery
     */
    struct Image *CheckMark;

    UBYTE *Title;                     /* the title text for this window */

    /* the Screen pointer is used only if you've defined a CUSTOMSCREEN and
     * want this Window to open in it.  If so, you pass the address of the
     * Custom Screen structure in this variable.  Otherwise, this variable
     * is ignored and doesn't have to be initialized.
     */
    struct Screen *Screen;

    /* WFLG_SUPER_BITMAP Window?  If so, put the address of your BitMap
     * structure in this variable.  If not, this variable is ignored and
     * doesn't have to be initialized
     */
    struct BitMap *BitMap;

    /* the values describe the minimum and maximum sizes of your Windows.
     * these matter only if you've chosen the WFLG_SIZEGADGET option,
     * which means that you want to let the User to change the size of
     * this Window.  You describe the minimum and maximum sizes that the
     * Window can grow by setting these variables.  You can initialize
     * any one these to zero, which will mean that you want to duplicate
     * the setting for that dimension (if MinWidth == 0, MinWidth will be
     * set to the opening Width of the Window).
     * You can change these settings later using SetWindowLimits().
     * If you haven't asked for a SIZING Gadget, you don't have to
     * initialize any of these variables.
     */
    WORD MinWidth, MinHeight;       /* minimums */
    UWORD MaxWidth, MaxHeight;       /* maximums */

    /* the type variable describes the Screen in which you want this Window to
     * open.  The type value can either be CUSTOMSCREEN or one of the
     * system standard Screen Types such as WBENCHSCREEN.  See the
     * type definitions under the Screen structure.
     */
    UWORD Type;

};

/* The following structure is the future NewWindow.  Compatibility
 * issues require that the size of NewWindow not change.
 * Data in the common part (NewWindow) indicates the the extension
 * fields are being used.
 * NOTE WELL: This structure may be subject to future extension.
 * Writing code depending on its size is not allowed.
 */
struct ExtNewWindow
{
```

```
    WORD LeftEdge, TopEdge;
    WORD Width, Height;

    UBYTE DetailPen, BlockPen;
    ULONG IDCMPFlags;
    ULONG Flags;
    struct Gadget *FirstGadget;

    struct Image *CheckMark;

    UBYTE *Title;
    struct Screen *Screen;
    struct BitMap *BitMap;

    WORD MinWidth, MinHeight;
    UWORD MaxWidth, MaxHeight;

    /* the type variable describes the Screen in which you want this Window to
     * open.  The type value can either be CUSTOMSCREEN or one of the
     * system standard Screen Types such as WBENCHSCREEN.  See the
     * type definitions under the Screen structure.
     * A new possible value for this field is PUBLICSCREEN, which
     * defines the window as a 'visitor' window.  See below for
     * additional information provided.
     */
    UWORD Type;

    /* ------------------------------------------------------- *
     * extensions for V36
     * if the NewWindow Flag value WFLG_NW_EXTENDED is set, then
     * this field is assumed to point to an array ( or chain of arrays)
     * of TagItem structures.  See also ExtNewScreen for another
     * use of TagItems to pass optional data.
     *
     * see below for tag values and the corresponding data.
     */
    struct TagItem      *Extension;
};

/*
 * The TagItem ID's (ti_Tag values) for OpenWindowTagList() follow.
 * They are values in a TagItem array passed as extension/replacement
 * values for the data in NewWindow.  OpenWindowTagList() can actually
 * work well with a NULL NewWindow pointer.
 */

#define WA_Dummy        (TAG_USER + 99) /* 0x80000063   */

/* these tags simply override NewWindow parameters */
#define WA_Left                 (WA_Dummy + 0x01)
#define WA_Top                  (WA_Dummy + 0x02)
#define WA_Width                (WA_Dummy + 0x03)
#define WA_Height               (WA_Dummy + 0x04)
#define WA_DetailPen            (WA_Dummy + 0x05)
#define WA_BlockPen             (WA_Dummy + 0x06)
#define WA_IDCMP                (WA_Dummy + 0x07)
                        /* "bulk" initialization of NewWindow.Flags */
#define WA_Flags                (WA_Dummy + 0x08)
#define WA_Gadgets              (WA_Dummy + 0x09)
#define WA_Checkmark            (WA_Dummy + 0x0A)
#define WA_Title                (WA_Dummy + 0x0B)
                        /* means you don't have to call SetWindowTitles
                         * after you open your window
                         */
#define WA_ScreenTitle          (WA_Dummy + 0x0C)
#define WA_CustomScreen         (WA_Dummy + 0x0D)
```

```
    */
    UWORD *Pointer;     /* sprite data */
    BYTE PtrHeight;     /* sprite height (not including sprite padding) */
    BYTE PtrWidth;      /* sprite width (must be less than or equal to 16) */
    BYTE XOffset, YOffset;      /* sprite offsets */

    /* the IDCMP Flags and User's and Intuition's Message Ports */
    ULONG IDCMPFlags;   /* User-selected flags */
    struct MsgPort *UserPort, *WindowPort;
    struct IntuiMessage *MessageKey;

    UBYTE DetailPen, BlockPen;  /* for bar/border/gadget rendering */

    /* the CheckMark is a pointer to the imagery that will be used when
     * rendering MenuItems of this Window that want to be checkmarked
     * if this is equal to NULL, you'll get the default imagery
     */
    struct Image *CheckMark;

    UBYTE *ScreenTitle; /* if non-null, Screen title when Window is active */

    /* These variables have the mouse coordinates relative to the
     * inner-Window of WFLG_GIMMEZEROZERO Windows.  This is compared with the
     * MouseX and MouseY variables, which contain the mouse coordinates
     * relative to the upper-left corner of the Window, WFLG_GIMMEZEROZERO
     * notwithstanding
     */
    WORD GZZMouseX;
    WORD GZZMouseY;
    /* these variables contain the width and height of the inner-Window of
     * WFLG_GIMMEZEROZERO Windows
     */
    WORD GZZWidth;
    WORD GZZHeight;

    UBYTE *ExtData;

    BYTE *UserData;     /* general-purpose pointer to User data extension */

    /** 11/18/85: this pointer keeps a duplicate of what
     * Window.RPort->Layer is _supposed_ to be pointing at
     */
    struct Layer *WLayer;

    /* NEW 1.2: need to keep track of the font that
     * OpenWindow opened, in case user SetFont's into RastPort
     */
    struct TextFont *IFont;

    /* (V36) another flag word (the Flags field is used up).
     * At present, all flag values are system private.
     * Until further notice, you may not change nor use this field.
     */
    ULONG       MoreFlags;

    /**** Data beyond this point are Intuition Private.  DO NOT USE ****/
};


/* --- Flags requested at OpenWindow() time by the application --------- */
#define WFLG_SIZEGADGET     0x00000001  /* include sizing system-gadget? */
#define WFLG_DRAGBAR        0x00000002  /* include dragging system-gadget? */
#define WFLG_DEPTHGADGET    0x00000004  /* include depth arrangement gadget? */
#define WFLG_CLOSEGADGET    0x00000008  /* include close-box system-gadget? */

#define WFLG_SIZEBRIGHT     0x00000010  /* size gadget uses right border */
```

```
#define WFLG_SIZEBBOTTOM    0x00000020  /* size gadget uses bottom border */

/* --- refresh modes ------------------------------------------------------ */
/* combinations of the WFLG_REFRESHBITS select the refresh type */
#define WFLG_REFRESHBITS    0x000000C0
#define WFLG_SMART_REFRESH  0x00000000
#define WFLG_SIMPLE_REFRESH 0x00000040
#define WFLG_SUPER_BITMAP   0x00000080
#define WFLG_OTHER_REFRESH  0x000000C0

#define WFLG_BACKDROP       0x00000100  /* this is a backdrop window */

#define WFLG_REPORTMOUSE    0x00000200  /* to hear about every mouse move */

#define WFLG_GIMMEZEROZERO  0x00000400  /* a GimmeZeroZero window       */

#define WFLG_BORDERLESS     0x00000800  /* to get a Window sans border */

#define WFLG_ACTIVATE       0x00001000  /* when Window opens, it's Active */


/* FLAGS SET BY INTUITION */
#define WFLG_WINDOWACTIVE   0x00002000  /* this window is the active one */
#define WFLG_INREQUEST      0x00004000  /* this window is in request mode */
#define WFLG_MENUSTATE      0x00008000  /* Window is active with Menus on */

/* --- Other User Flags --------------------------------------------------- */
#define WFLG_RMBTRAP        0x00010000  /* Catch RMB events for your own */
#define WFLG_NOCAREREFRESH  0x00020000  /* not to be bothered with REFRESH */

/* --- Other Intuition Flags ---------------------------------------------- */
#define WFLG_WINDOWREFRESH  0x01000000  /* Window is currently refreshing */
#define WFLG_WBENCHWINDOW   0x02000000  /* WorkBench tool ONLY Window */
#define WFLG_WINDOWTICKED   0x04000000  /* only one timer tick at a time */


/* - V36 new Flags which the programmer may specify in NewWindow.Flags  */
#define WFLG_NW_EXTENDED    0x00040000  /* extension data provided      */
                                        /* see struct ExtNewWindow      */

/* --- V36 Flags to be set only by Intuition -------------------------  */
#define WFLG_VISITOR        0x08000000  /* visitor window               */
#define WFLG_ZOOMED         0x10000000  /* identifies "zoom state"      */
#define WFLG_HASZOOM        0x20000000  /* windowhas a zoom gadget      */

/* --- Other Window Values ----------------------------------------------- */
#define DEFAULTMOUSEQUEUE       (5)     /* no more mouse messages       */

/* --- see struct IntuiMessage for the IDCMP Flag definitions ------------- */




/* ======================================================================== */
/* === NewWindow ========================================================== */
/* ======================================================================== */
/*
 * Note that the new extension fields have been removed.  Use ExtNewWindow
 * structure below to make use of these fields
 */
struct NewWindow
{
    WORD LeftEdge, TopEdge;             /* screen dimensions of window */
    WORD Width, Height;                 /* screen dimensions of window */

    UBYTE DetailPen, BlockPen;          /* for bar/border/gadget rendering */
    UWORD       Green;          /* only 6 bits recognized in V36       */
    UWORD       Blue;           /* only 6 bits recognized in V36       */
};

/* === Easy Requester Specification ====================================== */
/* see also autodocs for EasyRequest and BuildEasyRequest      */
/* NOTE: This structure may grow in size in the future         */
struct EasyStruct {
    ULONG       es_StructSize;  /* should be sizeof (struct EasyStruct )*/
    ULONG       es_Flags;       /* should be 0 for now                  */
    UBYTE       *es_Title;      /* title of requester window            */
    UBYTE       *es_TextFormat; /* 'printf' style formatting string     */
    UBYTE       *es_GadgetFormat; /* 'printf' style formatting string   */
};



/* ======================================================================== */
/* === Miscellaneous ====================================================== */
/* ======================================================================== */

/* = MACROS ============================================================== */
#define MENUNUM(n) (n & 0x1F)
#define ITEMNUM(n) ((n >> 5) & 0x003F)
#define SUBNUM(n) ((n >> 11) & 0x001F)

#define SHIFTMENU(n) (n & 0x1F)
#define SHIFTITEM(n) ((n & 0x3F) << 5)
#define SHIFTSUB(n) ((n & 0x1F) << 11)

#define FULLMENUNUM( menu, item, sub )  \
        ( SHIFTSUB(sub) | SHIFTITEM(item) | SHIFTMENU(menu) )

#define SRBNUM(n)     (0x08 - (n >> 4))  /* SerRWBits -> read bits per char */
#define SWBNUM(n)     (0x08 - (n & 0x0F))/* SerRWBits -> write bits per chr */
#define SSBNUM(n)     (0x01 + (n >> 4))  /* SerStopBuf -> stop bits per chr */
#define SPARNUM(n)    (n >> 4)           /* SerParShk -> parity setting    */
#define SHAKNUM(n)    (n & 0x0F) /* SerParShk -> handshake mode    */


/* = MENU STUFF =========================================================== */
#define NOMENU 0x001F
#define NOITEM 0x003F
#define NOSUB  0x001F
#define MENUNULL 0xFFFF


/* = =RJ='s peculiarities ================================================= */
#define FOREVER for(;;)
#define SIGN(x) ( ((x) > 0) - ((x) < 0) )
#define NOT !

/* these defines are for the COMMSEQ and CHECKIT menu stuff.  If CHECKIT,
 * I'll use a generic Width (for all resolutions) for the CheckMark.
 * If COMMSEQ, likewise I'll use this generic stuff
 */
#define CHECKWIDTH      19
#define COMMWIDTH       27
#define LOWCHECKWIDTH   13
#define LOWCOMMWIDTH    16


/* these are the AlertNumber defines.  if you are calling DisplayAlert()
 * the AlertNumber you supply must have the ALERT_TYPE bits set to one
 * of these patterns
 */
```

```
#define ALERT_TYPE      0x80000000
#define RECOVERY_ALERT  0x00000000      /* the system can recover from this */
#define DEADEND_ALERT   0x80000000      /* no recovery possible, this is it */


/* When you're defining IntuiText for the Positive and Negative Gadgets
 * created by a call to AutoRequest(), these defines will get you
 * reasonable-looking text.  The only field without a define is the IText
 * field; you decide what text goes with the Gadget
 */
#define AUTOFRONTPEN    0
#define AUTOBACKPEN     1
#define AUTODRAWMODE    JAM2
#define AUTOLEFTEDGE    6
#define AUTOTOPEDGE     3
#define AUTOITEXTFONT   NULL
#define AUTONEXTTEXT    NULL


/* --- RAWMOUSE Codes and Qualifiers (Console OR IDCMP) ------------------- */
#define SELECTUP        (IECODE_LBUTTON | IECODE_UP_PREFIX)
#define SELECTDOWN      (IECODE_LBUTTON)
#define MENUUP          (IECODE_RBUTTON | IECODE_UP_PREFIX)
#define MENUDOWN        (IECODE_RBUTTON)
#define MIDDLEDOWN      (IECODE_MBUTTON)
#define MIDDLEUP        (IECODE_MBUTTON | IECODE_UP_PREFIX)
#define ALTLEFT         (IEQUALIFIER_LALT)
#define ALTRIGHT        (IEQUALIFIER_RALT)
#define AMIGALEFT       (IEQUALIFIER_LCOMMAND)
#define AMIGARIGHT      (IEQUALIFIER_RCOMMAND)
#define AMIGAKEYS       (AMIGALEFT | AMIGARIGHT)

#define CURSORUP        0x4C
#define CURSORLEFT      0x4F
#define CURSORRIGHT     0x4E
#define CURSORDOWN      0x4D
#define KEYCODE_Q       0x10
#define KEYCODE_Z       0x31
#define KEYCODE_X       0x32
#define KEYCODE_V       0x34
#define KEYCODE_B       0x35
#define KEYCODE_N       0x36
#define KEYCODE_M       0x37
#define KEYCODE_LESS    0x38
#define KEYCODE_GREATER 0x39

/* Include obsolete identifiers: */
#ifndef INTUITION_IOBSOLETE_H
#include <intuition/iobsolete.h>
#endif

#endif
```

```c
#ifndef INTUITION_INTUITIONBASE_H
#define INTUITION_INTUITIONBASE_H 1
/*
**  $Filename: intuition/intuitionbase.h $
**  $Release: 2.04 $
**  $Revision: 36.19 $
**  $Date: 90/07/12 $
**
**  Public part of IntuitionBase structure and supporting structures
**
**  (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_TYPES_H
#include <exec/types.h>
#endif

#ifndef EXEC_LIBRARIES_H
#include <exec/libraries.h>
#endif

#ifndef INTUITION_INTUITION_H
#include <intuition/intuition.h>
#endif


#ifndef EXEC_INTERRUPTS_H
#include <exec/interrupts.h>
#endif

/* these are the display modes for which we have corresponding parameter
 *  settings in the config arrays
 */
#define DMODECOUNT      0x0002  /* how many modes there are */
#define HIRESPICK       0x0000
#define LOWRESPICK      0x0001

#define EVENTMAX 10             /* size of event array */

/* these are the system Gadget defines */
#define RESCOUNT        2
#define HIRESGADGET     0
#define LOWRESGADGET    1

#define GADGETCOUNT     8
#define UPFRONTGADGET   0
#define DOWNBACKGADGET  1
#define SIZEGADGET      2
#define CLOSEGADGET     3
#define DRAGGADGET      4
#define SUPFRONTGADGET  5
#define SDOWNBACKGADGET 6
#define SDRAGGADGET     7

/* ====================================================================== */
/* === IntuitionBase ==================================================== */
/* ====================================================================== */
/*
 * Be sure to protect yourself against someone modifying these data as
 * you look at them.  This is done by calling:
 *
 * lock = LockIBase(0), which returns a ULONG.  When done call
 * UnlockIBase(lock) where lock is what LockIBase() returned.
 */

```

```c
/* This structure is strictly READ ONLY */
struct IntuitionBase
{
    struct Library LibNode;

    struct View ViewLord;

    struct Window *ActiveWindow;
    struct Screen *ActiveScreen;

    /* the FirstScreen variable points to the frontmost Screen.  Screens are
     * then maintained in a front to back order using Screen.NextScreen
     */
    struct Screen *FirstScreen; /* for linked list of all screens */

    ULONG Flags;        /* values are all system private */
    WORD        MouseY, MouseX;
                        /* note "backwards" order of these             */

    ULONG Seconds;      /* timestamp of most current input event */
    ULONG Micros;       /* timestamp of most current input event */

    /* I told you this was private.
     * The data beyond this point has changed, is changing, and
     * will continue to change.
     */
};

#endif

```

```
#ifndef INTUITION_IOBSOLETE_H
#define INTUITION_IOBSOLETE_H

/*
**  $Filename: intuition/iobsolete.h $
**  $Release: 2.04 $
**  $Revision: 36.2 $
**  $Date: 90/11/02 $
**
**  Obsolete identifiers for Intuition.  Use the new ones instead!
**
**  (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/


/* This file contains:
 *
 * 1.  The traditional identifiers for gadget Flags, Activation, and Type,
 * and for window Flags and IDCMP classes.  They are defined in terms
 * of their new versions, which serve to prevent confusion between
 * similar-sounding but different identifiers (like IDCMP_WINDOWACTIVE
 * and WFLG_ACTIVATE).
 *
 * 2.  Some tag names and constants whose labels were adjusted after V36.
 *
 * 3.  Some tag names that were used only during the V36-beta cycle.
 *
 * By default, 1 and 2 are enabled, while 3 are excluded.
 *
 * #define INTUI_V36_NAMES_ONLY to exclude the traditional identifiers and
 * the original V36 names of some identifiers.
 *
 * #define INTUI_V36_BETA_NAMES to include the beta tag-names.
 */


#ifndef INTUITION_INTUITION_H
#include <intuition/intuition.h>
#endif

/* #define INTUI_V36_NAMES_ONLY to remove these older names */

#ifndef INTUI_V36_NAMES_ONLY


/* V34-style Gadget->Flags names: */

#define GADGHIGHBITS    GFLG_GADGHIGHBITS
#define GADGHCOMP       GFLG_GADGHCOMP
#define GADGHBOX        GFLG_GADGHBOX
#define GADGHIMAGE      GFLG_GADGHIMAGE
#define GADGHNONE       GFLG_GADGHNONE
#define GADGIMAGE       GFLG_GADGIMAGE
#define GRELBOTTOM      GFLG_RELBOTTOM
#define GRELRIGHT       GFLG_RELRIGHT
#define GRELWIDTH       GFLG_RELWIDTH
#define GRELHEIGHT      GFLG_RELHEIGHT
#define SELECTED        GFLG_SELECTED
#define GADGDISABLED    GFLG_DISABLED
#define LABELMASK       GFLG_LABELMASK
#define LABELITEXT      GFLG_LABELITEXT
#define LABELSTRING     GFLG_LABELSTRING
#define LABELIMAGE      GFLG_LABELIMAGE


```

```
/* V34-style Gadget->Activation flag names: */

#define RELVERIFY       GACT_RELVERIFY
#define GADGIMMEDIATE   GACT_IMMEDIATE
#define ENDGADGET       GACT_ENDGADGET
#define FOLLOWMOUSE     GACT_FOLLOWMOUSE
#define RIGHTBORDER     GACT_RIGHTBORDER
#define LEFTBORDER      GACT_LEFTBORDER
#define TOPBORDER       GACT_TOPBORDER
#define BOTTOMBORDER    GACT_BOTTOMBORDER
#define BORDERSNIFF     GACT_BORDERSNIFF
#define TOGGLESELECT    GACT_TOGGLESELECT
#define BOOLEXTEND      GACT_BOOLEXTEND
#define STRINGLEFT      GACT_STRINGLEFT
#define STRINGCENTER    GACT_STRINGCENTER
#define STRINGRIGHT     GACT_STRINGRIGHT
#define LONGINT         GACT_LONGINT
#define ALTKEYMAP       GACT_ALTKEYMAP
#define STRINGEXTEND    GACT_STRINGEXTEND
#define ACTIVEGADGET    GACT_ACTIVEGADGET


/* V34-style Gadget->Type names: */

#define GADGETTYPE      GTYP_GADGETTYPE
#define SYSGADGET       GTYP_SYSGADGET
#define SCRGADGET       GTYP_SCRGADGET
#define GZZGADGET       GTYP_GZZGADGET
#define REQGADGET       GTYP_REQGADGET
#define SIZING          GTYP_SIZING
#define WDRAGGING       GTYP_WDRAGGING
#define SDRAGGING       GTYP_SDRAGGING
#define WUPFRONT        GTYP_WUPFRONT
#define SUPFRONT        GTYP_SUPFRONT
#define WDOWNBACK       GTYP_WDOWNBACK
#define SDOWNBACK       GTYP_SDOWNBACK
#define CLOSE           GTYP_CLOSE
#define BOOLGADGET      GTYP_BOOLGADGET
#define GADGET0002      GTYP_GADGET0002
#define PROPGADGET      GTYP_PROPGADGET
#define STRGADGET       GTYP_STRGADGET
#define CUSTOMGADGET    GTYP_CUSTOMGADGET
#define GTYPEMASK       GTYP_GTYPEMASK


/* V34-style IDCMP class names: */

#define SIZEVERIFY      IDCMP_SIZEVERIFY
#define NEWSIZE         IDCMP_NEWSIZE
#define REFRESHWINDOW   IDCMP_REFRESHWINDOW
#define MOUSEBUTTONS    IDCMP_MOUSEBUTTONS
#define MOUSEMOVE       IDCMP_MOUSEMOVE
#define GADGETDOWN      IDCMP_GADGETDOWN
#define GADGETUP        IDCMP_GADGETUP
#define REQSET          IDCMP_REQSET
#define MENUPICK        IDCMP_MENUPICK
#define CLOSEWINDOW     IDCMP_CLOSEWINDOW
#define RAWKEY          IDCMP_RAWKEY
#define REQVERIFY       IDCMP_REQVERIFY
#define REQCLEAR        IDCMP_REQCLEAR
#define MENUVERIFY      IDCMP_MENUVERIFY
#define NEWPREFS        IDCMP_NEWPREFS
#define DISKINSERTED    IDCMP_DISKINSERTED
#define DISKREMOVED     IDCMP_DISKREMOVED
#define WBENCHMESSAGE   IDCMP_WBENCHMESSAGE
#define ACTIVEWINDOW    IDCMP_ACTIVEWINDOW
```

```
#define INACTIVEWINDOW  IDCMP_INACTIVEWINDOW
#define DELTAMOVE       IDCMP_DELTAMOVE
#define VANILLAKEY      IDCMP_VANILLAKEY
#define INTUITICKS      IDCMP_INTUITICKS
#define IDCMPUPDATE     IDCMP_IDCMPUPDATE
#define MENUHELP        IDCMP_MENUHELP
#define CHANGEWINDOW    IDCMP_CHANGEWINDOW
#define LONELYMESSAGE   IDCMP_LONELYMESSAGE


/* V34-style Window->Flags names: */

#define WINDOWSIZING    WFLG_SIZEGADGET
#define WINDOWDRAG      WFLG_DRAGBAR
#define WINDOWDEPTH     WFLG_DEPTHGADGET
#define WINDOWCLOSE     WFLG_CLOSEGADGET
#define SIZEBRIGHT      WFLG_SIZEBRIGHT
#define SIZEBBOTTOM     WFLG_SIZEBBOTTOM
#define REFRESHBITS     WFLG_REFRESHBITS
#define SMART_REFRESH   WFLG_SMART_REFRESH
#define SIMPLE_REFRESH  WFLG_SIMPLE_REFRESH
#define SUPER_BITMAP    WFLG_SUPER_BITMAP
#define OTHER_REFRESH   WFLG_OTHER_REFRESH
#define BACKDROP        WFLG_BACKDROP
#define REPORTMOUSE     WFLG_REPORTMOUSE
#define GIMMEZEROZERO   WFLG_GIMMEZEROZERO
#define BORDERLESS      WFLG_BORDERLESS
#define ACTIVATE        WFLG_ACTIVATE
#define WINDOWACTIVE    WFLG_WINDOWACTIVE
#define INREQUEST       WFLG_INREQUEST
#define MENUSTATE       WFLG_MENUSTATE
#define RMBTRAP         WFLG_RMBTRAP
#define NOCAREREFRESH   WFLG_NOCAREREFRESH
#define WINDOWREFRESH   WFLG_WINDOWREFRESH
#define WBENCHWINDOW    WFLG_WBENCHWINDOW
#define WINDOWTICKED    WFLG_WINDOWTICKED
#define NW_EXTENDED     WFLG_NW_EXTENDED
#define VISITOR         WFLG_VISITOR
#define ZOOMED          WFLG_ZOOMED
#define HASZOOM         WFLG_HASZOOM


/* These are the obsolete tag names for general gadgets, proportional gadgets,
 * and string gadgets.  Use the mixed-case equivalents from gadgetclass.h
 * instead.
 */

#define GA_LEFT                 GA_Left
#define GA_RELRIGHT             GA_RelRight
#define GA_TOP                  GA_Top
#define GA_RELBOTTOM            GA_RelBottom
#define GA_WIDTH                GA_Width
#define GA_RELWIDTH             GA_RelWidth
#define GA_HEIGHT               GA_Height
#define GA_RELHEIGHT            GA_RelHeight
#define GA_TEXT                 GA_Text
#define GA_IMAGE                GA_Image
#define GA_BORDER               GA_Border
#define GA_SELECTRENDER         GA_SelectRender
#define GA_HIGHLIGHT            GA_Highlight
#define GA_DISABLED             GA_Disabled
#define GA_GZZGADGET            GA_GZZGadget
#define GA_USERDATA             GA_UserData
#define GA_SPECIALINFO          GA_SpecialInfo
#define GA_SELECTED             GA_Selected
#define GA_ENDGADGET            GA_EndGadget
```

```
#define GA_IMMEDIATE            GA_Immediate
#define GA_RELVERIFY            GA_RelVerify
#define GA_FOLLOWMOUSE          GA_FollowMouse
#define GA_RIGHTBORDER          GA_RightBorder
#define GA_LEFTBORDER           GA_LeftBorder
#define GA_TOPBORDER            GA_TopBorder
#define GA_BOTTOMBORDER         GA_BottomBorder
#define GA_TOGGLESELECT         GA_ToggleSelect
#define GA_SYSGADGET            GA_SysGadget
#define GA_SYSGTYPE             GA_SysGType
#define GA_PREVIOUS             GA_Previous
#define GA_NEXT                 GA_Next
#define GA_DRAWINFO             GA_DrawInfo
#define GA_INTUITEXT            GA_IntuiText
#define GA_LABELIMAGE           GA_LabelImage

#define PGA_FREEDOM             PGA_Freedom
#define PGA_BORDERLESS          PGA_Borderless
#define PGA_HORIZPOT            PGA_HorizPot
#define PGA_HORIZBODY           PGA_HorizBody
#define PGA_VERTPOT             PGA_VertPot
#define PGA_VERTBODY            PGA_VertBody
#define PGA_TOTAL               PGA_Total
#define PGA_VISIBLE             PGA_Visible
#define PGA_TOP                 PGA_Top

#define LAYOUTA_LAYOUTOBJ       LAYOUTA_LayoutObj
#define LAYOUTA_SPACING         LAYOUTA_Spacing
#define LAYOUTA_ORIENTATION     LAYOUTA_Orientation


/* These are the obsolete tag names for image attributes.
 * Use the mixed-case equivalents from imageclass.h instead.
 */

#define IMAGE_ATTRIBUTES        (IA_Dummy)
#define IA_LEFT                 IA_Left
#define IA_TOP                  IA_Top
#define IA_WIDTH                IA_Width
#define IA_HEIGHT               IA_Height
#define IA_FGPEN                IA_FGPen
#define IA_BGPEN                IA_BGPen
#define IA_DATA                 IA_Data
#define IA_LINEWIDTH            IA_LineWidth
#define IA_PENS                 IA_Pens
#define IA_RESOLUTION           IA_Resolution
#define IA_APATTERN             IA_APattern
#define IA_APATSIZE             IA_APatSize
#define IA_MODE                 IA_Mode
#define IA_FONT                 IA_Font
#define IA_OUTLINE              IA_Outline
#define IA_RECESSED             IA_Recessed
#define IA_DOUBLEEMBOSS         IA_DoubleEmboss
#define IA_EDGESONLY            IA_EdgesOnly
#define IA_SHADOWPEN            IA_ShadowPen
#define IA_HIGHLIGHTPEN         IA_HighlightPen


/* These are the obsolete identifiers for the various DrawInfo pens.
 * Use the uppercase versions in screens.h instead.
 */

#define detailPen       DETAILPEN
#define blockPen        BLOCKPEN
#define textPen         TEXTPEN
#define shinePen        SHINEPEN
```

```
#define shadowPen       SHADOWPEN
#define hifillPen       FILLPEN
#define hifilltextPen   FILLTEXTPEN
#define backgroundPen   BACKGROUNDPEN
#define hilighttextPen  HIGHLIGHTTEXTPEN
#define numDrIPens      NUMDRIPENS


#endif /* !INTUI_V36_NAMES_ONLY */


/* #define INTUI_V36_BETA_NAMES to enable these tag names that were used
 * during the V36 beta period only.  New code should use the official
 * tags instead.
 */

#ifdef INTUI_V36_BETA_NAMES

/* These are the obsolete tag-names for the OpenWindowTags().
 * Use the WA_... equivalents from intuition.h instead.
 */

#define W_Dummy                 WA_Dummy
#define W_LEFT                  WA_Left
#define W_TOP                   WA_Top
#define W_WIDTH                 WA_Width
#define W_HEIGHT                WA_Height
#define W_DETAILPEN             WA_DetailPen
#define W_BLOCKPEN              WA_BlockPen
#define W_IDCMP                 WA_IDCMP
#define W_FLAGS                 WA_Flags
#define W_GADGETS               WA_Gadgets
#define W_CHECKMARK             WA_Checkmark
#define W_TITLE                 WA_Title
#define W_SCREENTITLE           WA_ScreenTitle
#define W_CUSTOMSCREEN          WA_CustomScreen
#define W_SUPERBITMAP           WA_SuperBitMap
#define W_MINWIDTH              WA_MinWidth
#define W_MINHEIGHT             WA_MinHeight
#define W_MAXWIDTH              WA_MaxWidth
#define W_MAXHEIGHT             WA_MaxHeight
#define W_INNERWIDTH            WA_InnerWidth
#define W_INNERHEIGHT           WA_InnerHeight
#define W_PUBSCNAME             WA_PubScreenName
#define W_PUBSC                 WA_PubScreen
#define W_PUBSCFALLBACK         WA_PubScreenFallBack
#define W_WINDOWNAME            WA_WindowName
#define W_COLORS                WA_Colors
#define W_ZOOM                  WA_Zoom
#define W_MOUSEQUEUE            WA_MouseQueue
#define W_BACKFILL              WA_BackFill
#define W_RPTQUEUE              WA_RptQueue
#define W_SIZEGADGET            WA_SizeGadget
#define W_DRAGBAR               WA_DragBar
#define W_DEPTHGADGET           WA_DepthGadget
#define W_CLOSEGADGET           WA_CloseGadget
#define W_BACKDROP              WA_Backdrop
#define W_REPORTMOUSE           WA_ReportMouse
#define W_NOCAREREFRESH         WA_NoCareRefresh
#define W_BORDERLESS            WA_Borderless
#define W_ACTIVATE              WA_Activate
#define W_RMBTRAP               WA_RMBTrap
#define W_WBENCHWINDOW          WA_WBenchWindow
#define W_SIMPLE_REFRESH        WA_SimpleRefresh
#define W_SMART_REFRESH         WA_SmartRefresh
#define W_SIZEBRIGHT            WA_SizeBRight
```

```
#define W_SIZEBBOTTOM           WA_SizeBBottom
#define W_AUTOADJUST            WA_AutoAdjust
#define W_GIMMEZEROZERO         WA_GimmeZeroZero


/* These are the obsolete tag-names for the OpenScreenTags().
 * Use the SA_... equivalents from screens.h instead.
 */

#define S_DUMMY         SA_Dummy
#define S_LEFT          SA_Left
#define S_TOP           SA_Top
#define S_WIDTH         SA_Width
#define S_HEIGHT        SA_Height
#define S_DEPTH         SA_Depth
#define S_DETAILPEN     SA_DetailPen
#define S_BLOCKPEN      SA_BlockPen
#define S_TITLE         SA_Title
#define S_COLORS        SA_Colors
#define S_ERRORCODE     SA_ErrorCode
#define S_FONT          SA_Font
#define S_SYSFONT       SA_SysFont
#define S_TYPE          SA_Type
#define S_BITMAP        SA_BitMap
#define S_PUBNAME       SA_PubName
#define S_PUBSIG        SA_PubSig
#define S_PUBTASK       SA_PubTask
#define S_DISPLAYID     SA_DisplayID
#define S_DCLIP         SA_DClip
#define S_STDDCLIP      SA_Overscan
#define S_MONITORNAME   SA_Obsolete1
#define S_SHOWTITLE     SA_ShowTitle
#define S_BEHIND        SA_Behind
#define S_QUIET         SA_Quiet
#define S_AUTOSCROLL    SA_AutoScroll


#endif /* INTUI_V36_BETA_NAMES */


#endif /* INTUITION_IOBSOLETE_H */
```

```
#ifndef INTUITION_PREFERENCES_H
#define INTUITION_PREFERENCES_H TRUE
/*
**  $Filename: intuition/preferences.h $
**  $Release: 2.04 $
**  $Revision: 36.6 $
**  $Date: 91/02/01 $
**
**  Structure definition for old-style preferences
**
**  (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_TYPES_H
#include <exec/types.h>
#endif

#ifndef DEVICES_TIMER_H
#include <devices/timer.h>
#endif

/* ======================================================================== */
/* === Preferences ======================================================== */
/* ======================================================================== */

/* these are the definitions for the printer configurations */
#define FILENAME_SIZE   30                      /* Filename size */

#define POINTERSIZE (1 + 16 + 1) * 2    /* Size of Pointer data buffer */

/* These defines are for the default font size.  These actually describe the
 * height of the defaults fonts.  The default font type is the topaz
 * font, which is a fixed width font that can be used in either
 * eighty-column or sixty-column mode.  The Preferences structure reflects
 * which is currently selected by the value found in the variable FontSize,
 * which may have either of the values defined below.  These values actually
 * are used to select the height of the default font.  By changing the
 * height, the resolution of the font changes as well.
 */
#define TOPAZ_EIGHTY 8
#define TOPAZ_SIXTY 9

struct Preferences
{
    /* the default font height */
    BYTE FontHeight;                    /* height for system default font  */

    /* constant describing what's hooked up to the port */
    UBYTE PrinterPort;                  /* printer port connection         */

    /* the baud rate of the port */
    UWORD BaudRate;                     /* baud rate for the serial port   */

    /* various timing rates */
    struct timeval KeyRptSpeed;         /* repeat speed for keyboard       */
    struct timeval KeyRptDelay;         /* Delay before keys repeat        */
    struct timeval DoubleClick;         /* Interval allowed between clicks */

    /* Intuition Pointer data */
    UWORD PointerMatrix[POINTERSIZE];   /* Definition of pointer sprite    */
    BYTE XOffset;                       /* X-Offset for active 'bit'       */
    BYTE YOffset;                       /* Y-Offset for active 'bit'       */
    UWORD color17;                      /***********************************/
    UWORD color18;                      /* Colours for sprite pointer      */
    UWORD color19;                      /***********************************/
```

```
    UWORD PointerTicks;                 /* Sensitivity of the pointer      */

    /* Workbench Screen colors */
    UWORD color0;                       /***********************************/
    UWORD color1;                       /*  Standard default colours       */
    UWORD color2;                       /*   Used in the Workbench         */
    UWORD color3;                       /***********************************/

    /* positioning data for the Intuition View */
    BYTE ViewXOffset;                   /* Offset for top lefthand corner  */
    BYTE ViewYOffset;                   /* X and Y dimensions              */
    WORD ViewInitX, ViewInitY;          /* View initial offset values      */

    BOOL EnableCLI;                     /* CLI availability switch */

    /* printer configurations */
    UWORD PrinterType;                  /* printer type            */
    UBYTE PrinterFilename[FILENAME_SIZE];/* file for printer       */

    /* print format and quality configurations */
    UWORD PrintPitch;                   /* print pitch                     */
    UWORD PrintQuality;                 /* print quality                   */
    UWORD PrintSpacing;                 /* number of lines per inch        */
    UWORD PrintLeftMargin;              /* left margin in characters       */
    UWORD PrintRightMargin;             /* right margin in characters      */
    UWORD PrintImage;                   /* positive or negative            */
    UWORD PrintAspect;                  /* horizontal or vertical          */
    UWORD PrintShade;                   /* b&w, half-tone, or color        */
    WORD PrintThreshold;                /* darkness ctrl for b/w dumps     */

    /* print paper descriptors */
    UWORD PaperSize;                    /* paper size                      */
    UWORD PaperLength;                  /* paper length in number of lines */
    UWORD PaperType;                    /* continuous or single sheet      */

    /* Serial device settings: These are six nibble-fields in three bytes */
    /* (these look a little strange so the defaults will map out to zero) */
    UBYTE   SerRWBits;   /* upper nibble = (8-number of read bits)      */
                         /* lower nibble = (8-number of write bits)     */
    UBYTE   SerStopBuf;  /* upper nibble = (number of stop bits - 1)    */
                         /* lower nibble = (table value for BufSize)    */
    UBYTE   SerParShk;   /* upper nibble = (value for Parity setting)   */
                         /* lower nibble = (value for Handshake mode)   */
    UBYTE   LaceWB;      /* if workbench is to be interlaced            */

    UBYTE   WorkName[FILENAME_SIZE];  /* temp file for printer          */

    BYTE    RowSizeChange;      /* affect NormalDisplayRows/Columns     */
    BYTE    ColumnSizeChange;

    UWORD    PrintFlags;        /* user preference flags */
    UWORD    PrintMaxWidth;     /* max width of printed picture in 10ths/in */
    UWORD    PrintMaxHeight;    /* max height of printed picture in 10ths/in */
    UBYTE    PrintDensity;      /* print density */
    UBYTE    PrintXOffset;      /* offset of printed picture in 10ths/inch */

    UWORD    wb_Width;          /* override default workbench width  */
    UWORD    wb_Height;         /* override default workbench height */
    UBYTE    wb_Depth;          /* override default workbench depth  */

    UBYTE    ext_size;          /* extension information -- do not touch! */
                            /* extension size in blocks of 64 bytes */
};


/* Workbench Interlace (use one bit) */
```

```
#define LACEWB                  (1<< 0)
#define LW_RESERVED     1               /* internal use only */

/* Enable_CLI   */
#define SCREEN_DRAG     (1<<14)
#define MOUSE_ACCEL     (1<<15)

/* PrinterPort */
#define PARALLEL_PRINTER 0x00
#define SERIAL_PRINTER  0x01

/* BaudRate */
#define BAUD_110        0x00
#define BAUD_300        0x01
#define BAUD_1200       0x02
#define BAUD_2400       0x03
#define BAUD_4800       0x04
#define BAUD_9600       0x05
#define BAUD_19200      0x06
#define BAUD_MIDI       0x07

/* PaperType */
#define FANFOLD 0x00
#define SINGLE          0x80

/* PrintPitch */
#define PICA            0x000
#define ELITE           0x400
#define FINE            0x800

/* PrintQuality */
#define DRAFT           0x000
#define LETTER          0x100

/* PrintSpacing */
#define SIX_LPI         0x000
#define EIGHT_LPI       0x200

/* Print Image */
#define IMAGE_POSITIVE  0x00
#define IMAGE_NEGATIVE  0x01

/* PrintAspect */
#define ASPECT_HORIZ    0x00
#define ASPECT_VERT     0x01

/* PrintShade */
#define SHADE_BW        0x00
#define SHADE_GREYSCALE 0x01
#define SHADE_COLOR     0x02

/* PaperSize */
#define US_LETTER       0x00
#define US_LEGAL        0x10
#define N_TRACTOR       0x20
#define W_TRACTOR       0x30
#define CUSTOM          0x40

/* PrinterType */
#define CUSTOM_NAME             0x00
#define ALPHA_P_101             0x01
#define BROTHER_15XL            0x02
#define CBM_MPS1000             0x03
#define DIAB_630                0x04
#define DIAB_ADV_D25            0x05
#define DIAB_C_150              0x06
```

```
#define EPSON                   0x07
#define EPSON_JX_80             0x08
#define OKIMATE_20              0x09
#define QUME_LP_20              0x0A
/* new printer entries, 3 October 1985 */
#define HP_LASERJET             0x0B
#define HP_LASERJET_PLUS        0x0C

/* Serial Input Buffer Sizes */
#define SBUF_512        0x00
#define SBUF_1024       0x01
#define SBUF_2048       0x02
#define SBUF_4096       0x03
#define SBUF_8000       0x04
#define SBUF_16000      0x05

/* Serial Bit Masks */
#define SREAD_BITS      0xF0 /* for SerRWBits   */
#define SWRITE_BITS     0x0F

#define SSTOP_BITS      0xF0 /* for SerStopBuf  */
#define SBUFSIZE_BITS   0x0F

#define SPARITY_BITS    0xF0 /* for SerParShk   */
#define SHSHAKE_BITS    0x0F

/* Serial Parity (upper nibble, after being shifted by
 * macro SPARNUM() )
 */
#define SPARITY_NONE     0
#define SPARITY_EVEN     1
#define SPARITY_ODD      2

/* Serial Handshake Mode (lower nibble, after masking using
 * macro SHANKNUM() )
 */
#define SHSHAKE_XON      0
#define SHSHAKE_RTS      1
#define SHSHAKE_NONE     2

/* new defines for PrintFlags */

#define CORRECT_RED         0x0001  /* color correct red shades */
#define CORRECT_GREEN       0x0002  /* color correct green shades */
#define CORRECT_BLUE        0x0004  /* color correct blue shades */

#define CENTER_IMAGE        0x0008  /* center image on paper */

#define IGNORE_DIMENSIONS   0x0000 /* ignore max width/height settings */
#define BOUNDED_DIMENSIONS  0x0010  /* use max width/height as boundaries */
#define ABSOLUTE_DIMENSIONS 0x0020  /* use max width/height as absolutes */
#define PIXEL_DIMENSIONS    0x0040  /* use max width/height as prt pixels */
#define MULTIPLY_DIMENSIONS 0x0080 /* use max width/height as multipliers */

#define INTEGER_SCALING     0x0100  /* force integer scaling */

#define ORDERED_DITHERING   0x0000 /* ordered dithering */
#define HALFTONE_DITHERING  0x0200  /* halftone dithering */
#define FLOYD_DITHERING     0x0400 /* Floyd-Steinberg dithering */

#define ANTI_ALIAS          0x0800 /* anti-alias image */
#define GREY_SCALE2         0x1000 /* for use with hi-res monitor */

/* masks used for checking bits */

#define CORRECT_RGB_MASK    (CORRECT_RED|CORRECT_GREEN|CORRECT_BLUE)
```

intuition/preferences.h

Page 5

```
#define DIMENSIONS_MASK     (BOUNDED_DIMENSIONS|ABSOLUTE_DIMENSIONS|PIXEL_DIMENS
    IONS|MULTIPLY_DIMENSIONS)
#define DITHERING_MASK      (HALFTONE_DITHERING|FLOYD_DITHERING)

#endif
```

intuition/screens.h

Page 1

```
#ifndef INTUITION_SCREENS_H
#define INTUITION_SCREENS_H TRUE
/*
**  $Filename: intuition/screens.h $
**  $Release: 2.04 $
**  $Revision: 36.35 $
**  $Date: 91/02/12 $
**
**  The Screen and NewScreen structures and attributes
**
**  (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_TYPES_H
#include <exec/types.h>
#endif

#ifndef GRAPHICS_GFX_H
#include <graphics/gfx.h>
#endif

#ifndef GRAPHICS_CLIP_H
#include <graphics/clip.h>
#endif

#ifndef GRAPHICS_VIEW_H
#include <graphics/view.h>
#endif

#ifndef GRAPHICS_RASTPORT_H
#include <graphics/rastport.h>
#endif

#ifndef GRAPHICS_LAYERS_H
#include <graphics/layers.h>
#endif

#ifndef UTILITY_TAGITEM_H
#include <utility/tagitem.h>
#endif

/*
 * NOTE:  intuition/iobsolete.h is included at the END of this file!
 */

/* ========================================================================== */
/* === DrawInfo =========================================================== */
/* ========================================================================== */

/* This is a packet of information for graphics rendering.  It originates
 * with a Screen, and is gotten using GetScreenDrawInfo( screen );
 */

/* If you find dri_Version >= DRI_VERSION, you know this structure
 * has at least the fields defined in this version of the include file
 */
#define RI_VERSION      (1)     /* obsolete, will be removed             */
#define DRI_VERSION     (1)

struct DrawInfo
{
    UWORD       dri_Version;    /* will be  DRI_VERSION                  */
    UWORD       dri_NumPens;    /* guaranteed to be >= numDrIPens        */
    UWORD       *dri_Pens;      /* pointer to pen array                  */

    struct TextFont     *dri_Font;      /* screen default font          */
    UWORD       dri_Depth;      /* (initial) depth of screen bitmap     */

    struct {        /* from DisplayInfo database for initial display mode */
        UWORD   X;
        UWORD   Y;
    }           dri_Resolution;

    ULONG       dri_Flags;              /* defined below                */
    ULONG       dri_Reserved[7];        /* avoid recompilation ;^)      */
};

#define DRIF_NEWLOOK    0x00000001      /* specified SA_Pens, full treatment */

/* rendering pen number indexes into DrawInfo.dri_Pens[]        */
#define DETAILPEN        (0x0000)       /* compatible Intuition rendering pens
  */
#define BLOCKPEN         (0x0001)       /* compatible Intuition rendering pens
  */
#define TEXTPEN          (0x0002)       /* text on background
  */
#define SHINEPEN         (0x0003)       /* bright edge on 3D objects
  */
#define SHADOWPEN        (0x0004)       /* dark edge on 3D objects
  */
#define FILLPEN          (0x0005)       /* active-window/selected-gadget fill
  */
#define FILLTEXTPEN      (0x0006)       /* text over FILLPEN
  */
#define BACKGROUNDPEN    (0x0007)       /* always color 0
  */
#define HIGHLIGHTTEXTPEN (0x0008)       /* special color text, on background
  */

#define NUMDRIPENS       (0x0009)


/* ====================================================================== */
/* === Screen =========================================================== */
/* ====================================================================== */
struct Screen
{
    struct Screen *NextScreen;          /* linked list of screens */
    struct Window *FirstWindow;         /* linked list Screen's Windows */

    WORD LeftEdge, TopEdge;             /* parameters of the screen */
    WORD Width, Height;                 /* parameters of the screen */

    WORD MouseY, MouseX;                /* position relative to upper-left */

    UWORD Flags;                        /* see definitions below */

    UBYTE *Title;                       /* null-terminated Title text */
    UBYTE *DefaultTitle;                /* for Windows without ScreenTitle */

    /* Bar sizes for this Screen and all Window's in this Screen */
    /* Note that BarHeight is one less than the actual menu bar
     * height.  We're going to keep this in V36 for compatibility,
     * although V36 artwork might use that extra pixel
     *
     * Also, the title bar height of a window is calculated from the
     * screen's WBorTop field, plus the font height, plus one.
     */
    BYTE BarHeight, BarVBorder, BarHBorder, MenuVBorder, MenuHBorder;
    BYTE WBorTop, WBorLeft, WBorRight, WBorBottom;

```

```
    struct TextAttr *Font;              /* this screen's default font      */

    /* the display data structures for this Screen */
    struct ViewPort ViewPort;           /* describing the Screen's display */
    struct RastPort RastPort;           /* describing Screen rendering     */
    struct BitMap BitMap;               /* extra copy of RastPort BitMap   */
    struct Layer_Info LayerInfo;        /* each screen gets a LayerInfo    */

    /* Only system gadgets may be attached to a screen.
     *  You get the standard system Screen Gadgets automatically
     */
    struct Gadget *FirstGadget;

    UBYTE DetailPen, BlockPen;          /* for bar/border/gadget rendering */

    /* the following variable(s) are maintained by Intuition to support the
     * DisplayBeep() color flashing technique
     */
    UWORD SaveColor0;

    /* This layer is for the Screen and Menu bars */
    struct Layer *BarLayer;

    UBYTE *ExtData;

    UBYTE *UserData;    /* general-purpose pointer to User data extension */

    /**** Data below this point are SYSTEM PRIVATE ****/
};


/* --- FLAGS SET BY INTUITION ------------------------------------------ */
/* The SCREENTYPE bits are reserved for describing various Screen types
 * available under Intuition.
 */
#define SCREENTYPE      0x000F  /* all the screens types available      */
/* --- the definitions for the Screen Type ---------------------------- */
#define WBENCHSCREEN    0x0001  /* identifies the Workbench screen      */
#define PUBLICSCREEN    0x0002  /* public shared (custom) screen        */
#define CUSTOMSCREEN    0x000F  /* original custom screens              */

#define SHOWTITLE       0x0010  /* this gets set by a call to ShowTitle() */

#define BEEPING         0x0020  /* set when Screen is beeping (private) */

#define CUSTOMBITMAP    0x0040  /* if you are supplying your own BitMap */

#define SCREENBEHIND    0x0080  /* if you want your screen to open behind
                                 * already open screens
                                 */
#define SCREENQUIET     0x0100  /* if you do not want Intuition to render
                                 * into your screen (gadgets, title)
                                 */
#define SCREENHIRES     0x0200  /* do not use lowres gadgets  (private) */

#define NS_EXTENDED     0x1000          /* ExtNewScreen.Extension is valid
 */
/* V36 applications can use OpenScreenTagList() instead of NS_EXTENDED  */

#define AUTOSCROLL      0x4000  /* screen is to autoscoll               */

#define STDSCREENHEIGHT -1      /* supply in NewScreen.Height           */
#define STDSCREENWIDTH -1       /* supply in NewScreen.Width            */

/*
 * Screen attribute tag ID's.  These are used in the ti_Tag field of
 * TagItem arrays passed to OpenScreenTagList() (or in the
 * ExtNewScreen.Extension field).
 */

/* Screen attribute tags.  Please use these versions, not those in
 * iobsolete.h.
 */

#define SA_Dummy        (TAG_USER + 32)
/*
 * these items specify items equivalent to fields in NewScreen
 */
#define SA_Left         (SA_Dummy + 0x0001)
#define SA_Top          (SA_Dummy + 0x0002)
#define SA_Width        (SA_Dummy + 0x0003)
#define SA_Height       (SA_Dummy + 0x0004)
                        /* traditional screen positions and dimensions  */
#define SA_Depth        (SA_Dummy + 0x0005)
                        /* screen bitmap depth                          */
#define SA_DetailPen    (SA_Dummy + 0x0006)
                        /* serves as default for windows, too           */
#define SA_BlockPen     (SA_Dummy + 0x0007)
#define SA_Title        (SA_Dummy + 0x0008)
                        /* default screen title                         */
#define SA_Colors       (SA_Dummy + 0x0009)
                        /* ti_Data is an array of struct ColorSpec,
                         * terminated by ColorIndex = -1.  Specifies
                         * initial screen palette colors.
                         */
#define SA_ErrorCode    (SA_Dummy + 0x000A)
                        /* ti_Data points to LONG error code (values below)*/
#define SA_Font         (SA_Dummy + 0x000B)
                        /* equiv. to NewScreen.Font                     */
#define SA_SysFont      (SA_Dummy + 0x000C)
                        /* Selects one of the preferences system fonts:
                         *      0 - old DefaultFont, fixed-width
                         *      1 - WB Screen preferred font
                         */
#define SA_Type         (SA_Dummy + 0x000D)
                        /* equiv. to NewScreen.Type                     */
#define SA_BitMap       (SA_Dummy + 0x000E)
                        /* ti_Data is pointer to custom BitMap.  This
                         * implies type of CUSTOMBITMAP
                         */
#define SA_PubName      (SA_Dummy + 0x000F)
                        /* presence of this tag means that the screen
                         * is to be a public screen.  Please specify
                         * BEFORE the two tags below
                         */
#define SA_PubSig       (SA_Dummy + 0x0010)
#define SA_PubTask      (SA_Dummy + 0x0011)
                        /* Task ID and signal for being notified that
                         * the last window has closed on a public screen.
                         */
#define SA_DisplayID    (SA_Dummy + 0x0012)
                        /* ti_Data is new extended display ID from
                         * <graphics/displayinfo.h>.
                         */
#define SA_DClip        (SA_Dummy + 0x0013)
                        /* ti_Data points to a rectangle which defines
                         * screen display clip region
                         */
#define SA_Overscan     (SA_Dummy + 0x0014)
                        /* was S_STDDCLIP.  Set to one of the OSCAN_
                         * specifiers below to get a system standard
                         * overscan region for your display clip,
```

```
                         * screen dimensions (unless otherwise specified),
                         * and automatically centered position (partial
                         * support only so far).
                         * If you use this, you shouldn't specify
                         * SA_DClip.  SA_Overscan is for "standard"
                         * overscan dimensions, SA_DClip is for
                         * your custom numeric specifications.
                         */
#define SA_Obsolete1    (SA_Dummy + 0x0015)
                        /* obsolete S_MONITORNAME                       */

/** booleans **/
#define SA_ShowTitle    (SA_Dummy + 0x0016)
                        /* boolean equivalent to flag SHOWTITLE         */
#define SA_Behind       (SA_Dummy + 0x0017)
                        /* boolean equivalent to flag SCREENBEHIND      */
#define SA_Quiet        (SA_Dummy + 0x0018)
                        /* boolean equivalent to flag SCREENQUIET       */
#define SA_AutoScroll   (SA_Dummy + 0x0019)
                        /* boolean equivalent to flag AUTOSCROLL        */
#define SA_Pens         (SA_Dummy + 0x001A)
                        /* pointer to ~0 terminated UWORD array, as
                         * found in struct DrawInfo
                         */
#define SA_FullPalette  (SA_Dummy + 0x001B)
                        /* boolean: initialize color table to entire
                         *  preferences palette (32 for V36), rather
                         * than compatible pens 0-3, 17-19, with
                         * remaining palette as returned by GetColorMap()
                         */

/* this is an obsolete tag included only for compatibility with V35
 * interim release for the A2024 and Viking monitors
 */
#ifndef NSTAG_EXT_VPMODE
#define NSTAG_EXT_VPMODE (TAG_USER | 1)
#endif


/* OpenScreen error codes, which are returned in the (optional) LONG
 * pointed to by ti_Data for the SA_ErrorCode tag item
 */
#define OSERR_NOMONITOR (1)     /* named monitor spec not available     */
#define OSERR_NOCHIPS   (2)     /* you need newer custom chips          */
#define OSERR_NOMEM     (3)     /* couldn't get normal memory           */
#define OSERR_NOCHIPMEM (4)     /* couldn't get chipmem                 */
#define OSERR_PUBNOTUNIQUE (5)  /* public screen name already used      */
#define OSERR_UNKNOWNMODE (6)   /* don't recognize mode asked for       */

/* ===================================================================== */
/* === NewScreen ======================================================= */
/* ===================================================================== */
/* note: to use the Extended field, you must use the
 * new ExtNewScreen structure, below
 */
struct NewScreen
{
    WORD LeftEdge, TopEdge, Width, Height, Depth;  /* screen dimensions */

    UBYTE DetailPen, BlockPen;  /* for bar/border/gadget rendering      */

    UWORD ViewModes;            /* the Modes for the ViewPort (and View) */

    UWORD Type;                 /* the Screen type (see defines above)  */

    struct TextAttr *Font;      /* this Screen's default text attributes */

    UBYTE *DefaultTitle;        /* the default title for this Screen    */

    struct Gadget *Gadgets;     /* UNUSED:  Leave this NULL             */

    /* if you are opening a CUSTOMSCREEN and already have a BitMap
     * that you want used for your Screen, you set the flags CUSTOMBITMAP in
     * the Type field and you set this variable to point to your BitMap
     * structure.  The structure will be copied into your Screen structure,
     * after which you may discard your own BitMap if you want
     */
    struct BitMap *CustomBitMap;
};

/*
 * For compatibility reasons, we need a new structure for extending
 * NewScreen.  Use this structure is you need to use the new Extension
 * field.
 *
 * NOTE: V36-specific applications should use the
 * OpenScreenTagList( newscreen, tags ) version of OpenScreen().
 * Applications that want to be V34-compatible as well may safely use the
 * ExtNewScreen structure.  Its tags will be ignored by V34 Intuition.
 *
 */
struct ExtNewScreen
{
    WORD LeftEdge, TopEdge, Width, Height, Depth;
    UBYTE DetailPen, BlockPen;
    UWORD ViewModes;
    UWORD Type;
    struct TextAttr *Font;
    UBYTE *DefaultTitle;
    struct Gadget *Gadgets;
    struct BitMap *CustomBitMap;

    struct TagItem      *Extension;
                                /* more specification data, scanned if
                                 * NS_EXTENDED is set in NewScreen.Type
                                 */
};

/* === Overscan Types ===       */
#define OSCAN_TEXT      (1)     /* entirely visible     */
#define OSCAN_STANDARD  (2)     /* just past edges      */
#define OSCAN_MAX       (3)     /* as much as possible  */
#define OSCAN_VIDEO     (4)     /* even more than is possible   */


/* === Public Shared Screen Node ===    */

/* This is the representative of a public shared screen.
 * This is an internal data structure, but some functions may
 * present a copy of it to the calling application.  In that case,
 * be aware that the screen pointer of the structure can NOT be
 * used safely, since there is no guarantee that the referenced
 * screen will remain open and a valid data structure.
 *
 * Never change one of these.
 */

struct PubScreenNode    {
    struct Node         psn_Node;       /* ln_Name is screen name */
    struct Screen       *psn_Screen;
    UWORD               psn_Flags;      /* below                */
    WORD                psn_Size;       /* includes name buffer */
```

```
    WORD                psn_VisitorCount; /* how many visitor windows */
    struct Task         *psn_SigTask;   /* who to signal when visitors gone */
    UBYTE               psn_SigBit;     /* which signal */
};

#define PSNF_PRIVATE    (0x0001)

#define MAXPUBSCREENNAME        (139)   /* names no longer, please      */

/* pub screen modes     */
#define SHANGHAI        0x0001  /* put workbench windows on pub screen */
#define POPPUBSCREEN    0x0002  /* pop pub screen to front when visitor opens */


/* Include obsolete identifiers: */
#ifndef INTUITION_IOBSOLETE_H
#include <intuition/iobsolete.h>
#endif

#endif
```

```
#ifndef INTUITION_SGHOOKS_H
#define INTUITION_SGHOOKS_H TRUE
/*
**  $Filename: intuition/sghooks.h $
**  $Release: 2.04 $
**  $Revision: 36.4 $
**  $Date: 91/02/12 $
**
**  string gadget extensions and hooks
**
**  (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_TYPES_H
#include <exec/types.h>
#endif

struct StringExtend {
    /* display specifications    */
    struct TextFont *Font;      /* must be an open Font (not TextAttr)  */
    UBYTE       Pens[2];        /* color of text/backgroun              */
    UBYTE       ActivePens[2];  /* colors when gadget is active         */

    /* edit specifications      */
    ULONG       InitialModes;   /* initial mode flags, below            */
    struct Hook *EditHook;      /* if non-NULL, must supply WorkBuffer  */
    UBYTE       *WorkBuffer;    /* must be as large as StringInfo.Buffer*/

    ULONG       Reserved[4];    /* set to 0                             */
};

struct SGWork   {
    /* set up when gadget is first activated    */
    struct Gadget       *Gadget;        /* the contestant itself        */
    struct StringInfo   *StringInfo;    /* easy access to sinfo         */
    UBYTE               *WorkBuffer;    /* intuition's planned result   */
    UBYTE               *PrevBuffer;    /* what was there before        */
    ULONG               Modes;          /* current mode                 */

    /* modified for each input event    */
    struct InputEvent   *IEvent;        /* actual event: do not change */
    UWORD               Code;           /* character code, if one byte */
    WORD                BufferPos;      /* cursor position              */
    WORD                NumChars;
    ULONG               Actions;        /* what Intuition will do       */
    LONG                LongInt;        /* temp storage for longint     */

    struct GadgetInfo   *GadgetInfo;    /* see cghooks.h                */
    UWORD               EditOp;         /* from constants below         */
};

/* SGWork.EditOp -
 * These values indicate what basic type of operation the global
 * editing hook has performed on the string before your gadget's custom
 * editing hook gets called.  You do not have to be concerned with the
 * value your custom hook leaves in the EditOp field, only if you
 * write a global editing hook.
 *
 * For most of these general edit operations, you'll want to compare
 * the BufferPos and NumChars of the StringInfo (before global editing)
 * and SGWork (after global editing).
 */

#define EO_NOOP         (0x0001)
        /* did nothing                                                  */
```

```
#define EO_DELBACKWARD  (0x0002)
        /* deleted some chars (maybe 0).                                */
#define EO_DELFORWARD   (0x0003)
        /* deleted some characters under and in front of the cursor    */
#define EO_MOVECURSOR   (0x0004)
        /* moved the cursor                                             */
#define EO_ENTER        (0x0005)
        /* "enter" or "return" key, terminate                           */
#define EO_RESET        (0x0006)
        /* current Intuition-style undo                                 */
#define EO_REPLACECHAR  (0x0007)
        /* replaced one character and (maybe) advanced cursor           */
#define EO_INSERTCHAR   (0x0008)
        /* inserted one char into string or added one at end            */
#define EO_BADFORMAT    (0x0009)
        /* didn't like the text data, e.g., Bad LONGINT                 */
#define EO_BIGCHANGE    (0x000A)        /* unused by Intuition  */
        /* complete or major change to the text, e.g. new string        */
#define EO_UNDO         (0x000B)        /* unused by Intuition  */
        /* some other style of undo                                     */
#define EO_CLEAR        (0x000C)
        /* clear the string                                             */
#define EO_SPECIAL      (0x000D)        /* unused by Intuition  */
        /* some operation that doesn't fit into the categories here     */


/* Mode Flags definitions (ONLY first group allowed as InitialModes)    */
#define SGM_REPLACE     (1L << 0)       /* replace mode                 */
/* please initialize StringInfo with in-range value of BufferPos
 * if you are using SGM_REPLACE mode.
 */

#define SGM_FIXEDFIELD  (1L << 1)       /* fixed length buffer          */
                                        /* always set SGM_REPLACE, too  */
#define SGM_NOFILTER    (1L << 2)       /* don't filter control chars   */

/* SGM_EXITHELP is new for V37, and ignored by V36: */
#define SGM_EXITHELP    (1L << 7)       /* exit with code = 0x5F if HELP hit */


/* These Mode Flags are for internal use only                           */
#define SGM_NOCHANGE    (1L << 3)       /* no edit changes yet          */
#define SGM_NOWORKB     (1L << 4)       /* Buffer == PrevBuffer         */
#define SGM_CONTROL     (1L << 5)       /* control char escape mode     */
#define SGM_LONGINT     (1L << 6)       /* an intuition longint gadget  */

/* String Gadget Action Flags (put in SGWork.Actions by EditHook)      */
#define SGA_USE         (0x1L)  /* use contents of SGWork               */
#define SGA_END         (0x2L)  /* terminate gadget, code in Code field */
#define SGA_BEEP        (0x4L)  /* flash the screen for the user        */
#define SGA_REUSE       (0x8L)  /* reuse input event                    */
#define SGA_REDISPLAY   (0x10L) /* gadget visuals changed               */

/* New for V37: */
#define SGA_NEXTACTIVE  (0x20L) /* Make next possible gadget active.    */
#define SGA_PREVACTIVE  (0x40L) /* Make previous possible gadget active.*/

/* function id for only existing custom string gadget edit hook */

#define SGH_KEY         (1L)    /* process editing keystroke            */
#define SGH_CLICK       (2L)    /* process mouse click cursor position  */

/* Here's a brief summary of how the custom string gadget edit hook works:
 *      You provide a hook in StringInfo.Extension.EditHook.
 *      The hook is called in the standard way with the 'object'
 *      a pointer to SGWork, and the 'message' a pointer to a command
 *      block, starting either with (longword) SGH_KEY, SGH_CLICK,
 *      or something new.
 *
 *      You return 0 if you don't understand the command (SGH_KEY is
 *      required and assumed).  Return non-zero if you implement the
 *      command.
 *
 *   SGH_KEY:
 *      There are no parameters following the command longword.
 *
 *      Intuition will put its idea of proper values in the SGWork
 *      before calling you, and if you leave SGA_USE set in the
 *      SGWork.Actions field, Intuition will use the values
 *      found in SGWork fields WorkBuffer, NumChars, BufferPos,
 *      and LongInt, copying the WorkBuffer back to the StringInfo
 *      Buffer.
 *
 *      NOTE WELL: You may NOT change other SGWork fields.
 *
 *      If you clear SGA_USE, the string gadget will be unchanged.
 *
 *      If you set SGA_END, Intuition will terminate the activation
 *      of the string gadget.  If you also set SGA_REUSE, Intuition
 *      will reuse the input event after it deactivates your gadget.
 *
 *      In this case, Intuition will put the value found in SGWork.Code
 *      into the IntuiMessage.Code field of the IDCMP_GADGETUP message it
 *      sends to the application.
 *
 *      If you set SGA_BEEP, Intuition will call DisplayBeep(); use
 *      this if the user has typed in error, or buffer is full.
 *
 *      Set SGA_REDISPLAY if the changes to the gadget warrant a
 *      gadget redisplay.  Note: cursor movement requires a redisplay.
 *
 *      Starting in V37, you may set SGA_PREVACTIVE or SGA_NEXTACTIVE
 *      when you set SGA_END.  This tells Intuition that you want
 *      the next or previous gadget with GFLG_TABCYCLE to be activated.
 *
 *   SGH_CLICK:
 *      This hook command is called when Intuition wants to position
 *      the cursor in response to a mouse click in the string gadget.
 *
 *      Again, here are no parameters following the command longword.
 *
 *      This time, Intuition has already calculated the mouse position
 *      character cell and put it in SGWork.BufferPos.  The previous
 *      BufferPos value remains in the SGWork.StringInfo.BufferPos.
 *
 *      Intuition will again use the SGWork fields listed above for
 *      SGH_KEY.  One restriction is that you are NOT allowed to set
 *      SGA_END or SGA_REUSE for this command.  Intuition will not
 *      stand for a gadget which goes inactive when you click in it.
 *
 *      You should always leave the SGA_REDISPLAY flag set, since Intuition
 *      uses this processing when activating a string gadget.
 */

#endif
```

```
#ifndef LIBRARIES_ASL_H
#define LIBRARIES_ASL_H 1

/*
**      $Filename: libraries/asl.h $
**      $Release: 2.04 $
**      $Revision: 36.4 $
**      $Date: 91/03/06 $
**
**      ASL library name and useful definitions.
**
**      (C) Copyright 1989,1990 Commodore-Amiga Inc. and Charlie Heath
**              All Rights Reserved
*/

#ifndef EXEC_TYPES_H
#include        <exec/types.h>
#endif

#ifndef EXEC_LISTS_H
#include        <exec/lists.h>
#endif

#ifndef EXEC_LIBRARIES_H
#include        <exec/libraries.h>
#endif

#ifndef UTILITY_HOOKS_H
#include <utility/hooks.h>
#endif

#ifndef UTILITY_TAGITEM_H
#include <utility/tagitem.h>
#endif

#ifndef WBArg
#include <workbench/startup.h>
#endif

#ifndef GRAPHICS_TEXT_H
#include <graphics/text.h>
#endif

/*
 ************************************************************************
 *      Standard definitions for asl library information               *
 ************************************************************************
 */

#define AslName         "asl.library"

/*
 ************************************************************************
 *                                                                      *
 *      The ASL file requester data structure...                        *
 *                                                                      *
 * The fields described here are for READ ACCESS to the structure       *
 * returned by AllocAslRequest( ASL_FileRequest, ... )                  *
 *                                                                      *
 * Any modifications MUST be done via TAGS either at the time of        *
 * creation by AllocAslRequest(), or when used, via AslRequest()        *
 *                                                                      *
 ************************************************************************
 */
struct FileRequester    {
        APTR    rf_Reserved1;
```

```
        BYTE    *rf_File;       /* Filename pointer             */
        BYTE    *rf_Dir;        /* Directory name pointer       */
        CPTR    rf_Reserved2;
        UBYTE   rf_Reserved3;
        UBYTE   rf_Reserved4;
        APTR    rf_Reserved5;
        WORD    rf_LeftEdge,rf_TopEdge; /* Preferred window pos */
        WORD    rf_Width,rf_Height;   /* Preferred window size  */
        WORD    rf_Reserved6;
        LONG    rf_NumArgs;     /* A-la WB Args, for multiselects */
        struct WBArg *rf_ArgList;
        APTR    rf_UserData;    /* Applihandle (you may write!!) */
        APTR    rf_Reserved7;
        APTR    rf_Reserved8;
        BYTE    *rf_Pat;        /* Pattern match pointer */
        };                      /* note - more reserved fields follow */


/*
 * The following defined values are the ASL_FuncFlags tag values which
 * are defined for the ASL file request.  These values may be passed
 * as a TagItem to modify the way the requester is presented.  Each
 * flag value defined has a description of the particular action.
 *
 * Also related to the ASL_FuncFlags values is the ASL_HookFunc tagitem,
 * which provides a callback function pointer to allow the application
 * to interact with the requester.  If an ASL_HookFunc TagItem is
 * provided, the hook function will be called like so:
 *
 * ULONG rf_Function(ULONG Mask, CPTR Object, CPTR AslRequester)
 *
 * The Mask value is a copy of the specific ASL_FuncFlags value
 * the callback is for; Object is a pointer to a data object.
 * AslRequester is a pointer to the requester structure.
 *
 * For the ASL file and font requesters, two ASL_FuncFlags values
 * are currently defined; FILF_DOWILDFUNC and FILF_DOMSGFUNC.
 *
 */

#define FILB_DOWILDFUNC 7L /* Called with an Object=AnchorPath, */
                           /*             ZERO return accepts.  */
#define FILB_DOMSGFUNC  6L /* Called with Object=IDCMP message  */
                           /*  for other window of shared port. */
                           /* You must return pointer to Object,*/
                           /* asl will reply the Object for you */
#define FILB_SAVE       5L /* For a SAVE operation, set this bit */
#define FILB_NEWIDCMP   4L /* Force a new IDCMP (only if rf_Window != NULL)  */
#define FILB_MULTISELECT 3L /* Request multiple selections returned from FR. */
                           /*   MULISELECT is ignored if FILB_SAVE is on     */
#define FILB_PATGAD     0L /* Ask for pattern gadget                         */


#define FILF_DOWILDFUNC (1L << FILB_DOWILDFUNC)
#define FILF_DOMSGFUNC  (1L << FILB_DOMSGFUNC)

#define FILF_SAVE       (1L << FILB_SAVE)
#define FILF_NEWIDCMP   (1L << FILB_NEWIDCMP)
#define FILF_MULTISELECT (1L << FILB_MULTISELECT)
#define FILF_PATGAD     (1L << FILB_PATGAD)


/* The following additional flags may be passed with the
 * ASL_ExtFlags1 tag.
 */
#define FIL1B_NOFILES   0L /* Do not want a file gadget, no files shown      */
```

libraries/asl.h Page 3

```
#define FIL1B_MATCHDIRS 1L /* Have Patgad or rf_Pat screen files AND DIRS      */

#define FIL1F_NOFILES    (1L << FIL1B_NOFILES)
#define FIL1F_MATCHDIRS (1L << FIL1B_MATCHDIRS)


/*
 ************************************************************************
 *      The ASL font requester data structure...                        *
 *                                                                      *
 * As with the FileRequest structure, the fields documented here are    *
 * for READ ACCESS ONLY.  Any modifications must be done via tags.      *
 ************************************************************************
 */
struct FontRequester    {
        APTR    fo_Reserved1[2];
        struct TextAttr fo_Attr;        /* Returned TextAttr             */
        UBYTE   fo_FrontPen;            /* Returned pens, if selected    */
        UBYTE   fo_BackPen;
        UBYTE   fo_DrawMode;
        APTR    fo_UserData;
        };

/* Bit defines for ASL_FuncFlags, for FONT requester                    */
/* See descriptive text for FILF_ values above for an overview.         */
/* Note - old mixed-case defines were nonstandard, now obsolete         */

#define FONB_FRONTCOLOR 0       /* Display Front color selector?        */
#define FONB_BACKCOLOR  1       /* Display Back color selector?         */
#define FONB_STYLES     2       /* Display Styles checkboxes?           */
#define FONB_DRAWMODE   3       /* Display DrawMode NWay?               */
#define FONB_FIXEDWIDTH 4       /* Only allow fixed-width fonts?        */
#define FONB_NEWIDCMP   5       /* Create a new IDCMP port, not shared  */
#define FONB_DOMSGFUNC  6       /* Called with Object=IntuiMessage for  */
                                /*      other windows in shared port,   */
                                /*   you must return Object pointer     */
                                /*   and asl will reply Object for you  */
#define FONB_DOWILDFUNC 7       /* Called with Object=TextAttr to approve*/
                                /*      NON-Zero return accepts         */

#define FONF_FRONTCOLOR (1L << FONB_FRONTCOLOR)
#define FONF_BACKCOLOR  (1L << FONB_BACKCOLOR)
#define FONF_STYLES     (1L << FONB_STYLES)
#define FONF_DRAWMODE   (1L << FONB_DRAWMODE)
#define FONF_FIXEDWIDTH (1L << FONB_FIXEDWIDTH)
#define FONF_NEWIDCMP   (1L << FONB_NEWIDCMP)
#define FONF_DOMSGFUNC  (1L << FONB_DOMSGFUNC)
#define FONF_DOWILDFUNC (1L << FONB_DOWILDFUNC)


/*************************************************************************/
/* Arguments to AllocAslRequest()                                        */
/* Types of requester structures which may be allocated:                 */
/*************************************************************************/
#define ASL_FileRequest 0
#define ASL_FontRequest 1

/*************************************************************************/
/* Tags for AllocAslRequest() and AslRequest()                           */
/*************************************************************************/

#define ASL_Dummy (TAG_USER + 0x80000)

#define ASL_Hail        ASL_Dummy+1     /* Hailing text follows            */
#define ASL_Window      ASL_Dummy+2     /* Parent window for IDCMP & screen */
#define ASL_LeftEdge    ASL_Dummy+3     /* Initialize LeftEdge             */
```

libraries/asl.h Page 4

```
#define ASL_TopEdge     ASL_Dummy+4     /* Initialize TopEdge              */
#define ASL_Width       ASL_Dummy+5
#define ASL_Height      ASL_Dummy+6
#define ASL_HookFunc    ASL_Dummy+7     /* Hook function pointer           */

/* Tags specific to file request                                         */
#define ASL_File        ASL_Dummy+8     /* Initial name of file follows    */
#define ASL_Dir         ASL_Dummy+9     /* Initial string of filerequest dir */

/* Tags specific to font request                                         */
#define ASL_FontName    ASL_Dummy+10    /* Initial font name               */
#define ASL_FontHeight  ASL_Dummy+11    /* Initial font height             */
#define ASL_FontStyles  ASL_Dummy+12    /* Initial font styles             */
#define ASL_FontFlags   ASL_Dummy+13    /* Initial font flags for textattr */
#define ASL_FrontPen    ASL_Dummy+14    /* Initial frontpen color          */
#define ASL_BackPen     ASL_Dummy+15    /* Initial backpen color           */
#define ASL_MinHeight   ASL_Dummy+16    /* Minimum font height to display  */
#define ASL_MaxHeight   ASL_Dummy+17    /* Max font height to display      */

#define ASL_OKText      ASL_Dummy+18    /* Text displayed in OK gadget     */
#define ASL_CancelText  ASL_Dummy+19    /* Text displayed in CANCEL gadget */
#define ASL_FuncFlags   ASL_Dummy+20    /* Function flags, depend on request */

#define ASL_ModeList    ASL_Dummy+21    /* Substitute list for font drawmodes */
#define ASL_ExtFlags1   ASL_Dummy+22    /* For passing extended FIL1F flags */

#define ASL_Pattern     ASL_FontName    /* File requester pattern string   */

/******** END of ASL Tag values ******************************************/

#endif
```

```
#ifndef LIBRARIES_COMMODITIES_H
#define LIBRARIES_COMMODITIES_H

/*
**   $Filename: libraries/commodities.h $
**   $Release: 2.04 $
**   $Revision: 36.1 $
**   $Date: 90/05/01 $
**
**   Commodities definitions.
**
**   (C) Copyright 1988,1989,1990 Commodore-Amiga Inc.
**       All Rights Reserved
*/

#ifndef EXEC_TYPES_H
#include <exec/types.h>
#endif

/*************************
 * object creation macros
 *************************/
#define CxFilter(d)            CreateCxObj((LONG)CX_FILTER, (LONG) d, 0)
#define CxTypeFilter(type)     CreateCxObj((LONG)CX_TYPEFILTER, (LONG) type, 0)
#define CxSender(port,id)      CreateCxObj((LONG)CX_SEND, (LONG) port, (LONG) id)
#define CxSignal(task,sig)     CreateCxObj((LONG)CX_SIGNAL,(LONG) task, (LONG) sig)
#define CxTranslate(ie)        CreateCxObj((LONG)CX_TRANSLATE, (LONG) ie, 0)
#define CxDebug(id)            CreateCxObj((LONG)CX_DEBUG, (LONG) id, 0)
#define CxCustom(action,id)    CreateCxObj((LONG)CX_CUSTOM,(LONG)action,(LONG)id)

/***************
 * Broker stuff
 ***************/

/* buffer sizes   */
#define CBD_NAMELEN      24
#define CBD_TITLELEN     40
#define CBD_DESCRLEN     40

/* CxBroker errors   */
#define CBERR_OK         0        /* No error                        */
#define CBERR_SYSERR     1        /* System error , no memory, etc   */
#define CBERR_DUP        2        /* uniqueness violation            */
#define CBERR_VERSION    3        /* didn't understand nb_VERSION    */

#define NB_VERSION       5        /* Version of NewBroker structure  */

struct NewBroker {
    BYTE     nb_Version;  /* set to NB_VERSION               */
    BYTE     *nb_Name;
    BYTE     *nb_Title;
    BYTE     *nb_Descr;
    SHORT    nb_Unique;
    SHORT    nb_Flags;
    BYTE     nb_Pri;
    /* new in V5   */
    struct MsgPort   *nb_Port;
    WORD     nb_ReservedChannel;  /* plans for later port sharing     */
};

/* Flags for nb_Unique */
#define NBU_DUPLICATE    0
#define NBU_UNIQUE       1        /* will not allow duplicates       */
#define NBU_NOTIFY       2        /* sends CXM_UNIQUE to existing broker */

/* Flags for nb_Flags */
```

```
#define   COF_SHOW_HIDE 4

/********
 * cxusr
 ********/

/** Fake data types for system private objects   */
#ifndef CX_H
typedef LONG   CxObj;
typedef LONG   CxMsg;
#endif

/* Pointer to Function returning Long   */
typedef LONG   (*PFL)();

/********************************/
/** Commodities Object Types   **/
/********************************/
#define CX_INVALID      0     /* not a valid object (probably null)  */
#define CX_FILTER       1     /* input event messages only           */
#define CX_TYPEFILTER   2     /* filter on message type       */
#define CX_SEND 3     /* sends a message                     */
#define CX_SIGNAL       4     /* sends a signal               */
#define CX_TRANSLATE    5     /* translates IE into chain            */
#define CX_BROKER       6     /* application representative          */
#define CX_DEBUG        7     /* dumps kprintf to serial port        */
#define CX_CUSTOM       8     /* application provids function        */
#define CX_ZERO 9     /* system terminator node       */

/*****************/
/** CxMsg types **/
/*****************/
#define CXM_UNIQUE   (1 << 4) /* sent down broker by CxBroker()      */
/* Obsolete: subsumed by CXM_COMMAND (below)   */

/* Messages of this type rattle around the Commodities input network.
 * They will be sent to you by a Sender object, and passed to you
 * as a synchronous function call by a Custom object.
 *
 * The message port or function entry point is stored in the object,
 * and the ID field of the message will be set to what you arrange
 * issuing object.
 *
 * The Data field will point to the input event triggering the
 * message.
 */
#define CXM_IEVENT   (1 << 5)

/* These messages are sent to a port attached to your Broker.
 * They are sent to you when the controller program wants your
 * program to do something.  The ID field identifies the command.
 *
 * The Data field will be used later.
 */
#define CXM_COMMAND   (1 << 6)

/* ID values   */
#define CXCMD_DISABLE    (15)  /* please disable yourself       */
#define CXCMD_ENABLE     (17)  /* please enable yourself        */
#define CXCMD_APPEAR     (19)  /* open your window, if you can  */
#define CXCMD_DISAPPEAR  (21)  /* go dormant                    */
#define CXCMD_KILL       (23)  /* go away for good              */
#define CXCMD_UNIQUE     (25)  /* someone tried to create a broker
                                * with your name.  Suggest you Appear.
                                */
#define CXCMD_LIST_CHG   (27)  /* Used by Exchange program. Someone */
```

```
                                 /* has changed the broker list       */

/* return values for BrokerCommand(): */
#define CMDE_OK (0)
#define CMDE_NOBROKER   (-1)
#define CMDE_NOPORT     (-2)
#define CMDE_NOMEM      (-3)

/* IMPORTANT NOTE: for V5:
 * Only CXM_IEVENT messages are passed through the input network.
 *
 * Other types of messages are sent to an optional port in your broker.
 *
 * This means that you must test the message type in your message handling,
 * if input messages and command messages come to the same port.
 *
 * Older programs have no broker port, so processing loops which
 * make assumptions about type won't encounter the new message types.
 *
 * The TypeFilter CxObject is hereby obsolete.
 *
 * It is less convenient for the application, but eliminates testing
 * for type of input messages.
 */

/**********************************************************/
/** CxObj Error Flags (return values from CxObjError())  **/
/**********************************************************/
#define COERR_ISNULL       1  /* you called CxError(NULL)          */
#define COERR_NULLATTACH   2  /* someone attached NULL to my list   */
#define COERR_BADFILTER    4  /* a bad filter description was given */
#define COERR_BADTYPE      8  /* unmatched type-specific operation  */


/******************************/
/* Input Expression structure */
/******************************/

#define IX_VERSION   2

struct InputXpression {
   UBYTE   ix_Version;     /* must be set to IX_VERSION  */
   UBYTE   ix_Class;       /* class must match exactly   */

   UWORD   ix_Code;        /* Bits that we want  */

   UWORD   ix_CodeMask;    /* Set bits here to indicate  */
                           /* which bits in ix_Code are  */
                                           /* don't care bits.
   */

   UWORD   ix_Qualifier;   /* Bits that we want  */

   UWORD   ix_QualMask;    /* Set bits here to indicate  */
                           /* which bits in ix_Qualifier */
                                           /* are don't care bits
   */

   UWORD   ix_QualSame;    /* synonyms in qualifier      */
   };

typedef struct InputXpression IX;

/* QualSame identifiers */
#define IXSYM_SHIFT  1     /* left- and right- shift are equivalent     */
#define IXSYM_CAPS   2     /* either shift or caps lock are equivalent  */
```

```
#define IXSYM_ALT    4     /* left- and right- alt are equivalent       */

/* corresponding QualSame masks */
#define IXSYM_SHIFTMASK (IEQUALIFIER_LSHIFT | IEQUALIFIER_RSHIFT)
#define IXSYM_CAPSMASK  (IXSYM_SHIFTMASK    | IEQUALIFIER_CAPSLOCK)
#define IXSYM_ALTMASK   (IEQUALIFIER_LALT   | IEQUALIFIER_RALT)

#define IX_NORMALQUALS  0x7FFF;/* for QualMask field: avoid RELATIVEMOUSE */

/* matches nothing   */
#define NULL_IX(I)   ((I)->ix_Class == IECLASS_NULL)

#endif
```

```
#ifndef LIBRARIES_CONFIGREGS_H
#define LIBRARIES_CONFIGREGS_H
/*
**      $Filename: libraries/configregs.h $
**      $Release: 2.04 $
**      $Revision: 36.12 $
**      $Date: 90/06/19 $
**
**      AutoConfig (tm) hardware register and bit definitions
**
**      (C) Copyright 1985,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/


#ifndef EXEC_TYPES_H
#include "exec/types.h"
#endif  /* EXEC_TYPES_H */


/*
** AutoConfig (tm) boards each contain a 32 byte "ExpansionRom" area that is
** read by the system software at configuration time.  Configuration of each
** board starts when the ConfigIn* signal is passed from the previous board
** (or from the system for the first board).  Each board will present it's
** ExpansionRom structure at location $00E80000 to be read by the system.
** This file defines the appearance of the ExpansionRom area.
**
** Expansion boards are actually organized such that only one nybble per
** 16 bit word contains valid information.  The low nybbles of each
** word are combined to fill the structure below. (This table is structured
** as LOGICAL information.  This means that it never corresponds exactly
** with a physical implementation.)
**
** The ExpansionRom space is further split into two regions:  The first 16
** bytes are read-only.  Except for the er_type field, this area is inverted
** by the system software when read in.  The second 16 bytes contain the
** control portion, where all read/write registers are located.
**
** The system builds one "ConfigDev" structure for each board found.  The
** list of boards can be examined using the expansion.library/FindConfigDev
** function.
**
** A special "hacker" Manufacturer ID number is reserved for test use:
** 2011 ($7DB).  When inverted this will look like $F824.
*/

struct ExpansionRom {           /* -First 16 bytes of the expansion ROM */
    UBYTE       er_Type;        /* Board type, size and flags */
    UBYTE       er_Product;     /* Product number, assigned by manufacturer */
    UBYTE       er_Flags;       /* Flags */
    UBYTE       er_Reserved03;  /* Must be zero ($ff inverted) */
    UWORD       er_Manufacturer; /* Unique ID,ASSIGNED BY COMMODORE-AMIGA! */
    ULONG       er_SerialNumber; /* Available for use by manufacturer */
    UWORD       er_InitDiagVec; /* Offset to optional "DiagArea" structure */
    UBYTE       er_Reserved0c;
    UBYTE       er_Reserved0d;
    UBYTE       er_Reserved0e;
    UBYTE       er_Reserved0f;
};


/*
** Note that use of the ec_BaseAddress register is tricky.  The system
** will actually write twice.  First the low order nybble is written
** to the ec_BaseAddress register+2 (D15-D12).  Then the entire byte is
```

```
** written to ec_BaseAddress (D15-D8).  This allows writing of a byte-wide
** address to nybble size registers.
*/

struct ExpansionControl {       /* -Second 16 bytes of the expansion ROM */
    UBYTE       ec_Interrupt;   /* Optional interrupt control register */
    UBYTE       ec_Z3_HighBase; /* Zorro III   : Config address bits 24-31 */
    UBYTE       ec_BaseAddress; /* Zorro II/III: Config address bits 16-23 */
    UBYTE       ec_Shutup;      /* The system writes here to shut up a board */
    UBYTE       ec_Reserved14;
    UBYTE       ec_Reserved15;
    UBYTE       ec_Reserved16;
    UBYTE       ec_Reserved17;
    UBYTE       ec_Reserved18;
    UBYTE       ec_Reserved19;
    UBYTE       ec_Reserved1a;
    UBYTE       ec_Reserved1b;
    UBYTE       ec_Reserved1c;
    UBYTE       ec_Reserved1d;
    UBYTE       ec_Reserved1e;
    UBYTE       ec_Reserved1f;
};

/*
** many of the constants below consist of a triplet of equivalent
** definitions: xxMASK is a bit mask of those bits that matter.
** xxBIT is the starting bit number of the field.  xxSIZE is the
** number of bits that make up the definition.  This method is
** used when the field is larger than one bit.
**
** If the field is only one bit wide then the xxB_xx and xxF_xx convention
** is used (xxB_xx is the bit number, and xxF_xx is mask of the bit).
*/

/* manifest constants */
#define E_SLOTSIZE              0x10000
#define E_SLOTMASK              0xffff
#define E_SLOTSHIFT             16

/* these define the free regions of Zorro memory space.
** THESE MAY WELL CHANGE FOR FUTURE PRODUCTS!
*/
#define E_EXPANSIONBASE 0x00e80000      /* Zorro II  config address */
#define EZ3_EXPANSIONBASE       0xff000000      /* Zorro III config address */

#define E_EXPANSIONSIZE 0x00080000      /* Zorro II  I/O type cards */
#define E_EXPANSIONSLOTS        8

#define E_MEMORYBASE            0x00200000      /* Zorro II  8MB space */
#define E_MEMORYSIZE            0x00800000
#define E_MEMORYSLOTS           128

#define EZ3_CONFIGAREA          0x40000000      /* Zorro III space */
#define EZ3_CONFIGAREAEND       0x7FFFFFFF      /* Zorro III space */
#define EZ3_SIZEGRANULARITY     0x00080000      /* 512K increments */



/**** er_Type definitions (ttldcmmm) ***************************************/

/* er_Type board type bits -- the OS ignores "old style" boards */
#define ERT_TYPEMASK            0xc0    ;Bits 7-6
#define ERT_TYPEBIT             6
#define ERT_TYPESIZE            2
#define ERT_NEWBOARD            0xc0
#define ERT_ZORROII             ERT_NEWBOARD
```

```
#define ERT_ZORROIII            0x80

/* other bits defined in er_Type */
#define ERTB_MEMLIST            5   /* Link RAM into free memory list */
#define ERTB_DIAGVALID          4   /* ROM vector is valid */
#define ERTB_CHAINEDCONFIG      3   /* Next config is part of the same card */

#define ERTF_MEMLIST            (1<<5)
#define ERTF_DIAGVALID          (1<<4)
#define ERTF_CHAINEDCONFIG      (1<<3)

/* er_Type field memory size bits */
#define ERT_MEMMASK             0x07    ;Bits 2-0
#define ERT_MEMBIT              0
#define ERT_MEMSIZE             3



/**** er_Flags byte -- for those things that didn't fit into the type byte ****/
/**** the hardware stores this byte in inverted form                      ****/
#define ERFF_MEMSPACE           (1<<7)  /* Wants to be in 8 meg space. */
#define ERFB_MEMSPACE           7       /* (NOT IMPLEMENTED) */

#define ERFF_NOSHUTUP           (1<<6)  /* Board can't be shut up */
#define ERFB_NOSHUTUP           6

#define ERFF_EXTENDED           (1<<5)  /* Zorro III: Use extended size table */
#define ERFB_EXTENDED           5       /*            for bits 0-2 of er_Type */
                                        /* Zorro II : Must be 0 */

#define ERFF_ZORRO_III          (1<<4)  /* Zorro III: must be 1 */
#define ERFB_ZORRO_III          4       /* Zorro II : must be 0 */

#define ERT_Z3_SSMASK           0x0F    /* Bits 3-0.  Zorro III Sub-Size.  How *
/
#define ERT_Z3_SSBIT            0       /* much space the card actually uses   *
/
#define ERT_Z3_SSSIZE           4       /* (regardless of config granularity)  *
/
                                        /* Zorro II : must be 0        */


/* ec_Interrupt register (unused) ********************************************/
#define ECIB_INTENA             1
#define ECIB_RESET              3
#define ECIB_INT2PEND           4
#define ECIB_INT6PEND           5
#define ECIB_INT7PEND           6
#define ECIB_INTERRUPTING       7

#define ECIF_INTENA             (1<<1)
#define ECIF_RESET              (1<<3)
#define ECIF_INT2PEND           (1<<4)
#define ECIF_INT6PEND           (1<<5)
#define ECIF_INT7PEND           (1<<6)
#define ECIF_INTERRUPTING       (1<<7)



/* figure out amount of memory needed by this box/board */
#define ERT_MEMNEEDED(t)        \
        (((t)&ERT_MEMMASK)? 0x10000 << (((t)&ERT_MEMMASK) -1) : 0x800000 )

/* same as ERT_MEMNEEDED, but return number of slots */
#define ERT_SLOTSNEEDED(t)      \
        (((t)&ERT_MEMMASK)? 1 << (((t)&ERT_MEMMASK)-1) : 0x80 )
```

```



/* convert a expansion slot number into a memory address */
#define EC_MEMADDR(slot)                ((slot) << (E_SLOTSHIFT) )

/* a kludge to get the byte offset of a structure */
#define EROFFSET(er)    ((int)&((struct ExpansionRom *)0)->er)
#define ECOFFSET(ec)    \
 (sizeof(struct ExpansionRom)+((int)&((struct ExpansionControl *)0)->ec))



/***************************************************************************
**
** these are the specifications for the diagnostic area.  If the Diagnostic
** Address Valid bit is set in the Board Type byte (the first byte in
** expansion space) then the Diag Init vector contains a valid offset.
**
** The Diag Init vector is actually a word offset from the base of the
** board.  The resulting address points to the base of the DiagArea
** structure.  The structure may be physically implemented either four,
** eight, or sixteen bits wide.  The code will be copied out into
** ram first before being called.
**
** The da_Size field, and both code offsets (da_DiagPoint and da_BootPoint)
** are offsets from the diag area AFTER it has been copied into ram, and
** "de-nibbleized" (if needed).  (In other words, the size is the size of
** the actual information, not how much address space is required to
** store it.)
**
** All bits are encoded with uninverted logic (e.g. 5 volts on the bus
** is a logic one).
**
** If your board is to make use of the boot facility then it must leave
** its config area available even after it has been configured.  Your
** boot vector will be called AFTER your board's final address has been
** set.
**
****************************************************************************/

struct DiagArea {
    UBYTE       da_Config;      /* see below for definitions */
    UBYTE       da_Flags;       /* see below for definitions */
    UWORD       da_Size;        /* the size (in bytes) of the total diag area */
    UWORD       da_DiagPoint;   /* where to start for diagnostics, or zero */
    UWORD       da_BootPoint;   /* where to start for booting */
    UWORD       da_Name;        /* offset in diag area where a string */
                                /*   identifier can be found (or zero if no */
                                /*   identifier is present). */

    UWORD       da_Reserved01;  /* two words of reserved data.  must be zero. */
    UWORD       da_Reserved02;
};

/* da_Config definitions */
/*
** DAC_BYTEWIDE can be simulated using DAC_NIBBLEWIDE.
*/
#define DAC_BUSWIDTH    0xC0 /* two bits for bus width */
#define DAC_NIBBLEWIDE  0x00
#define DAC_BYTEWIDE    0x40 /* BUG: Will not work under V34 Kickstart! */
#define DAC_WORDWIDE    0x80

#define DAC_BOOTTIME    0x30    /* two bits for when to boot */
#define DAC_NEVER       0x00    /* obvious */
```

libraries/configregs.h

Page 5

```
#define DAC_CONFIGTIME  0x10    /* call da_BootPoint when first configing */
                                /*   the device */
#define DAC_BINDTIME    0x20    /* run when binding drivers to boards */

/*
**
** These are the calling conventions for the diagnostic callback
** (from da_DiagPoint):
**
** A7 -- points to at least 2K of stack
** A6 -- ExecBase
** A5 -- ExpansionBase
** A3 -- your board's ConfigDev structure
** A2 -- Base of diag/init area that was copied
** A0 -- Base of your board
**
** Your board must return a value in D0.  If this value is NULL, then
** the diag/init area that was copied in will be returned to the free
** memory pool.
*/


#endif /* LIBRARIES_CONFIGREGS_H */
```

libraries/configvars.h

Page 1

```
#ifndef LIBRARIES_CONFIGVARS_H
#define LIBRARIES_CONFIGVARS_H
/*
**      $Filename: libraries/configvars.h $
**      $Release: 2.04 $
**      $Revision: 36.12 $
**      $Date: 90/11/22 $
**
**      Software structures used by AutoConfig (tm) boards
**
**      (C) Copyright 1985,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/
#ifndef EXEC_TYPES_H
#include "exec/types.h"
#endif  /* EXEC_TYPES_H */

#ifndef EXEC_NODES_H
#include "exec/nodes.h"
#endif /* EXEC_NODES_H */

#ifndef LIBRARIES_CONFIGREGS_H
#include "libraries/configregs.h"
#endif /* LIBRARIES_CONFIGREGS_H */

/* At early system startup time, one ConfigDev structure is created for
** each board found in the system.  Software may seach for ConfigDev
** structures by vendor & product ID number.  For debugging and diagnostic
** use, the entire list can be accessed.  See the expansion.library document
** for more information.
*/
struct ConfigDev {
    struct Node         cd_Node;
    UBYTE               cd_Flags;       /* (read/write) */
    UBYTE               cd_Pad;         /* reserved */
    struct ExpansionRom cd_Rom;         /* copy of board's expansion ROM */
    APTR                cd_BoardAddr; /* where in memory the board was placed */
    ULONG               cd_BoardSize;   /* size of board in bytes */
    UWORD               cd_SlotAddr;    /* which slot number (PRIVATE) */
    UWORD               cd_SlotSize;    /* number of slots (PRIVATE) */
    APTR                cd_Driver;      /* pointer to node of driver */
    struct ConfigDev *  cd_NextCD;      /* linked list of drivers to config */
    ULONG               cd_Unused[4];   /* for whatever the driver wants */
};

/* cd_Flags */
#define CDB_SHUTUP      0       /* this board has been shut up */
#define CDB_CONFIGME    1       /* this board needs a driver to claim it */
#define CDB_BADMEMORY   2       /* this board contains bad memory */

#define CDF_SHUTUP      0x01
#define CDF_CONFIGME    0x02
#define CDF_BADMEMORY   0x04

/* Boards are usually "bound" to software drivers.
** This structure is used by GetCurrentBinding() and SetCurrentBinding()
*/
struct CurrentBinding {
    struct ConfigDev *  cb_ConfigDev;           /* first configdev in chain */
    UBYTE *             cb_FileName;            /* file name of driver */
    UBYTE *             cb_ProductString;       /* product # string */
    UBYTE **            cb_ToolTypes;           /* tooltypes from disk object */
};

#endif /* LIBRARIES_CONFIGVARS_H */
```

libraries/diskfont.h Page 1

```
#ifndef LIBRARIES_DISKFONT_H
#define LIBRARIES_DISKFONT_H
/*
**      $Filename: libraries/diskfont.h $
**      $Release: 2.04 $
**      $Revision: 36.6 $
**      $Date: 90/11/26 $
**
**      diskfont library definitions
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef     EXEC_TYPES_H
#include "exec/types.h"
#endif
#ifndef     EXEC_NODES_H
#include "exec/nodes.h"
#endif
#ifndef     EXEC_LISTS_H
#include "exec/lists.h"
#endif
#ifndef     GRAPHICS_TEXT_H
#include "graphics/text.h"
#endif

#define     MAXFONTPATH 256   /* including null terminator */

struct FontContents {
    char    fc_FileName[MAXFONTPATH];
    UWORD   fc_YSize;
    UBYTE   fc_Style;
    UBYTE   fc_Flags;
};

struct TFontContents {
    char    tfc_FileName[MAXFONTPATH-2];
    UWORD   tfc_TagCount;       /* including the TAG_DONE tag */
    /*
     *  if tfc_TagCount is non-zero, tfc_FileName is overlayed with
     *  Text Tags starting at:  (struct TagItem *)
     *      &tfc_FileName[MAXFONTPATH-(tfc_TagCount*sizeof(struct TagItem))]
     */
    UWORD   tfc_YSize;
    UBYTE   tfc_Style;
    UBYTE   tfc_Flags;
};


#define  FCH_ID         0x0f00  /* FontContentsHeader, then FontContents */
#define  TFCH_ID        0x0f02  /* FontContentsHeader, then TFontContents */

struct FontContentsHeader {
    UWORD   fch_FileID;         /* FCH_ID */
    UWORD   fch_NumEntries;     /* the number of FontContents elements */
    /* struct FontContents fch_FC[], or struct TFontContents fch_TFC[]; */
};


#define  DFH_ID         0x0f80
#define  MAXFONTNAME    32      /* font name including ".font\0" */

struct DiskFontHeader {
    /* the following 8 bytes are not actually considered a part of the  */
    /* DiskFontHeader, but immediately preceed it. The NextSegment is    */
```

libraries/diskfont.h Page 2

```
    /* supplied by the linker/loader, and the ReturnCode is the code    */
    /* at the beginning of the font in case someone runs it...          */
    /*   ULONG dfh_NextSegment;                      \* actually a BPTR  */
    /*   ULONG dfh_ReturnCode;                       \* MOVEQ #0,D0 : RTS */
    /* here then is the official start of the DiskFontHeader...         */
    struct Node dfh_DF;         /* node to link disk fonts */
    UWORD   dfh_FileID;         /* DFH_ID */
    UWORD   dfh_Revision;       /* the font revision */
    LONG    dfh_Segment;        /* the segment address when loaded */
    char    dfh_Name[MAXFONTNAME]; /* the font name (null terminated) */
    struct TextFont dfh_TF;     /* loaded TextFont structure */
};

/* unfortunately, this needs to be explicitly typed */
/* used only if dfh_TF.tf_Style FSB_TAGGED bit is set */
#define dfh_TagList     dfh_Segment     /* destroyed during loading */


#define     AFB_MEMORY  0
#define     AFF_MEMORY  0x0001
#define     AFB_DISK    1
#define     AFF_DISK    0x0002
#define     AFB_SCALED  2
#define     AFF_SCALED  0x0004

#define     AFB_TAGGED  16      /* return TAvailFonts */
#define     AFF_TAGGED  0x10000L

struct AvailFonts {
    UWORD   af_Type;            /* MEMORY, DISK, or SCALED */
    struct TextAttr af_Attr;    /* text attributes for font */
};

struct TAvailFonts {
    UWORD   taf_Type;           /* MEMORY, DISK, or SCALED */
    struct TTextAttr taf_Attr;  /* text attributes for font */
};

struct AvailFontsHeader {
    UWORD   afh_NumEntries;      /* number of AvailFonts elements */
    /* struct AvailFonts afh_AF[], or struct TAvailFonts afh_TAF[]; */
};

#endif  /* LIBRARIES_DISKFONT_H */
```

libraries/dos.h

Page 1

```
#ifndef LIBRARIES_DOS_H
#define LIBRARIES_DOS_H
/*
**      $Filename: libraries/dos.h $
**      $Release: 2.04 $
**      $Revision: 36.2 $
**      $Date: 90/07/12 $
**
**      Standard C header for AmigaDOS
**
**      (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef DOS_DOS_H
#include "dos/dos.h"
#endif

#endif /* LIBRARIES_DOS_H */
```

---

libraries/dosextens.h

Page 1

```
#ifndef LIBRARIES_DOSEXTENS_H
#define LIBRARIES_DOSEXTENS_H
/*
**      $Filename: libraries/dosextens.h $
**      $Release: 2.04 $
**      $Revision: 36.2 $
**      $Date: 90/07/12 $
**
**      DOS structures not needed for the casual AmigaDOS user
**
**      (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef DOS_DOSEXTENS_H
#include "dos/dosextens.h"
#endif

#endif /* LIBRARIES_DOSEXTENS_H */
```

libraries/expansion.h

```
#ifndef LIBRARIES_EXPANSION_H
#define LIBRARIES_EXPANSION_H
/*
**      $Filename: libraries/expansion.h $
**      $Release: 2.04 $
**      $Revision: 36.7 $
**      $Date: 90/05/28 $
**
**      External definitions for expansion.library
**
**      (C) Copyright 1985,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#define EXPANSIONNAME   "expansion.library"

/* flags for the AddDosNode() call */
#define ADNB_STARTPROC  0
#define ADNF_STARTPROC  (1L<<0)

#endif /* LIBRARIES_EXPANSION_H */
```

libraries/expansionbase.h

```
#ifndef LIBRARIES_EXPANSIONBASE_H
#define LIBRARIES_EXPANSIONBASE_H
/*
**      $Filename: libraries/expansionbase.h $
**      $Release: 2.04 $
**      $Revision: 36.14 $
**      $Date: 91/02/13 $
**
**      Definitions for the expansion library base
**
**      (C) Copyright 1987,1988,1989,1991 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_TYPES_H
#include "exec/types.h"
#endif  /* EXEC_TYPES_H */

#ifndef EXEC_LIBRARIES_H
#include "exec/libraries.h"
#endif  /* EXEC_LIBRARIES_H */

#ifndef EXEC_SEMAPHORES_H
#include "exec/semaphores.h"
#endif  /* EXEC_SEMAPHORES_H */

#ifndef LIBRARIES_CONFIGVARS_H
#include "libraries/configvars.h"
#endif  /* LIBRARIES_CONFIGVARS_H */


/* BootNodes are scanned by dos.library at startup.  Items found on the
   list are started by dos. BootNodes are added with the AddDosNode() or
   the V36 AddBootNode() calls. */
struct BootNode
{
        struct Node bn_Node;
        UWORD   bn_Flags;
        APTR    bn_DeviceNode;
};


/* expansion.library has functions to manipulate most of the information in
   ExpansionBase.  Direct access is not permitted.  Use FindConfigDev()
   to scan the board list. */
struct  ExpansionBase
{
        struct Library  LibNode;
        UBYTE   Flags;                          /* read only (see below) */
        UBYTE   eb_Private01;                   /* private */
        ULONG   eb_Private02;                   /* private */
        ULONG   eb_Private03;                   /* private */
        struct  CurrentBinding  eb_Private04;   /* private */
        struct  List    eb_Private05;           /* private */
        struct  List    MountList;      /* contains struct BootNode entries */
        /* private */
};

/* error codes */
#define EE_OK           0
#define EE_LASTBOARD    40  /* could not shut him up */
#define EE_NOEXPANSION  41  /* not enough expansion mem; board shut up */
#define EE_NOMEMORY     42  /* not enough normal memory */
#define EE_NOBOARD      43  /* no board at that address */
#define EE_BADMEM       44  /* tried to add bad memory card */

```

libraries/expansionbase.h

```
/* Flags */
#define EBB_CLOGGED     0       /* someone could not be shutup */
#define EBF_CLOGGED     (1<<0)
#define EBB_SHORTMEM    1       /* ran out of expansion mem */
#define EBF_SHORTMEM    (1<<1)
#define EBB_BADMEM      2       /* tried to add bad memory card */
#define EBF_BADMEM      (1<<2)
#define EBB_DOSFLAG     3       /* reserved for use by AmigaDOS */
#define EBF_DOSFLAG     (1<<3)
#define EBB_KICKBACK33  4       /* reserved for use by AmigaDOS */
#define EBF_KICKBACK33  (1<<4)
#define EBB_KICKBACK36  5       /* reserved for use by AmigaDOS */
#define EBF_KICKBACK36  (1<<5)
/* If the following flag is set by a floppy's bootblock code, the initial
   open of the initial shell window will be delayed until the first output
   to that shell.  Otherwise the 1.3 compatible behavior applies. */
#define EBB_SILENTSTART 6
#define EBF_SILENTSTART (1<<6)


#endif  /* LIBRARIES_EXPANSIONBASE_H */
```

libraries/filehandler.h

```
#ifndef LIBRARIES_FILEHANDLER_H
#define LIBRARIES_FILEHANDLER_H
/*
**      $Filename: libraries/filehandler.h $
**      $Release: 2.04 $
**      $Revision: 36.2 $
**      $Date: 90/07/12 $
**
**      device and file handler specific code for AmigaDOS
**
**      (C) Copyright 1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef DOS_FILEHANDLER_H
#include "dos/filehandler.h"
#endif

#endif /* LIBRARIES_FILEHANDLER_H */
```

libraries/gadtools.h Page 1

```
#ifndef LIBRARIES_GADTOOLS_H
#define LIBRARIES_GADTOOLS_H
/*
**      $Filename: libraries/gadtools.h $
**      $Release: 2.04 $
**      $Revision: 36.14 $
**      $Date: 91/01/28 $
**
**      gadtools.library definitions
**
**      (C) Copyright 1989, 1990, Commodore-Amiga, Inc.
**      All Rights Reserved
*/

/*------------------------------------------------------------------------*/

#ifndef EXEC_TYPES_H
#include <exec/types.h>
#endif

#ifndef UTILITY_TAGITEM_H
#include <utility/tagitem.h>
#endif

#ifndef INTUITION_INTUITION_H
#include <intuition/intuition.h>
#endif

/*------------------------------------------------------------------------*/

/*  The kinds (almost classes) of gadgets in the toolkit.  Use these
    identifiers when calling CreateGadgetA() */

#define GENERIC_KIND    0
#define BUTTON_KIND     1
#define CHECKBOX_KIND   2
#define INTEGER_KIND    3
#define LISTVIEW_KIND   4
#define MX_KIND         5
#define NUMBER_KIND     6
#define CYCLE_KIND      7
#define PALETTE_KIND    8
#define SCROLLER_KIND   9
/* Kind number 10 is reserved */
#define SLIDER_KIND     11
#define STRING_KIND     12
#define TEXT_KIND       13

#define NUM_KINDS       14

/*------------------------------------------------------------------------*/

/*  These two definitions are obsolete, but are here for backwards
    compatibility.  You never need to worry about these: */
#define GADTOOLBIT      (0x8000)
/*  Use this mask to isolate the user part: */
#define GADTOOLMASK     (~GADTOOLBIT)

/*------------------------------------------------------------------------*/

/*  'Or' the appropriate set together for your Window IDCMPFlags: */

#define ARROWIDCMP      (IDCMP_GADGETUP | IDCMP_GADGETDOWN |\
        IDCMP_INTUITICKS | IDCMP_MOUSEBUTTONS)

#define BUTTONIDCMP     (IDCMP_GADGETUP)
```

libraries/gadtools.h Page 2

```
#define CHECKBOXIDCMP   (IDCMP_GADGETUP)
#define INTEGERIDCMP    (IDCMP_GADGETUP)
#define LISTVIEWIDCMP   (IDCMP_GADGETUP | IDCMP_GADGETDOWN |\
        IDCMP_MOUSEMOVE | ARROWIDCMP)

#define MXIDCMP         (IDCMP_GADGETDOWN)
#define NUMBERIDCMP     (NULL)
#define CYCLEIDCMP      (IDCMP_GADGETUP)
#define PALETTEIDCMP    (IDCMP_GADGETUP)

/*  Use ARROWIDCMP|SCROLLERIDCMP if your scrollers have arrows: */
#define SCROLLERIDCMP   (IDCMP_GADGETUP | IDCMP_GADGETDOWN | IDCMP_MOUSEMOVE)
#define SLIDERIDCMP     (IDCMP_GADGETUP | IDCMP_GADGETDOWN | IDCMP_MOUSEMOVE)
#define STRINGIDCMP     (IDCMP_GADGETUP)

#define TEXTIDCMP       (NULL)

/*------------------------------------------------------------------------*/

/*  Typical suggested spacing between "elements": */
#define INTERWIDTH      8
#define INTERHEIGHT     4

/*------------------------------------------------------------------------*/

/*  Generic NewGadget used by several of the gadget classes: */

struct NewGadget
    {
    WORD ng_LeftEdge, ng_TopEdge;       /*  gadget position */
    WORD ng_Width, ng_Height;           /*  gadget size */
    UBYTE *ng_GadgetText;               /*  gadget label */
    struct TextAttr *ng_TextAttr;       /*  desired font for gadget label */
    UWORD ng_GadgetID;                  /*  gadget ID */
    ULONG ng_Flags;                     /*  see below */
    APTR ng_VisualInfo;                 /*  Set to retval of GetVisualInfo() */
    APTR ng_UserData;                   /*  gadget UserData */
    };


/*  ng_Flags control certain aspects of the gadget.  The first five control
    the placement of the descriptive text.  All larger groups supply a
    default: */

#define PLACETEXT_LEFT  0x0001  /* Right-align text on left side */
#define PLACETEXT_RIGHT 0x0002  /* Left-align text on right side */
#define PLACETEXT_ABOVE 0x0004  /* Center text above */
#define PLACETEXT_BELOW 0x0008  /* Center text below */
#define PLACETEXT_IN    0x0010  /* Center text on */

#define NG_HIGHLABEL    0x0020  /* Highlight the label */

/*------------------------------------------------------------------------*/

/* Fill out an array of these and pass that to CreateMenus(): */

struct NewMenu
    {
    UBYTE nm_Type;              /*  See below */
    STRPTR nm_Label;            /*  Menu's label */
    STRPTR nm_CommKey;          /*  MenuItem Command Key Equiv */
    UWORD nm_Flags;             /*  Menu or MenuItem flags (see note) */
    LONG nm_MutualExclude;      /*  MenuItem MutualExclude word */
    APTR nm_UserData;           /*  For your own use, see note */
    };
```

```
/*  Each nm_Type should be one of these: */
#define NM_TITLE        1
#define NM_ITEM         2
#define NM_SUB          3
#define NM_END          0

#define MENU_IMAGE      128
#define IM_ITEM         (NM_ITEM | MENU_IMAGE)
#define IM_SUB          (NM_SUB | MENU_IMAGE)

/*  If you set your label to NM_BARLABEL, you'll get a separator bar. */
#define NM_BARLABEL     ((STRPTR)-1)


/*  The nm_Flags field is used to fill out either the Menu->Flags or
    MenuItem->Flags field.  Note that the sense of the MENUENABLED or
    ITEMENABLED bit is inverted between this use and Intuition's use,
    in other words, NewMenus are enabled by default.  The following
    labels are provided to disable them: */
#define NM_MENUDISABLED MENUENABLED
#define NM_ITEMDISABLED ITEMENABLED

/*  The following are pre-cleared (COMMSEQ, ITEMTEXT, and HIGHxxx are set
    later as appropriate): */
#define NM_FLAGMASK     (~(COMMSEQ | ITEMTEXT | HIGHFLAGS))

/*  You may choose among CHECKIT, MENUTOGGLE, and CHECKED.
    Toggle-select menuitems are of type CHECKIT|MENUTOGGLE, along
    with CHECKED if currently selected.  Mutually exclusive ones
    are of type CHECKIT, and possibly CHECKED too.  The nm_MutualExclude
    is a bit-wise representation of the items excluded by this one,
    so in the simplest case (choose 1 among n), these flags would be
    ~1, ~2, ~4, ~8, ~16, etc.  See the Intuition Menus chapter. */

/*  A UserData pointer can be associated with each Menu and MenuItem structure.
    The CreateMenus() call allocates space for a UserData after each
    Menu or MenuItem (header, item or sub-item).  You should use the
    GTMENU_USERDATA() or GTMENUITEM_USERDATA() macro to extract it. */

#define GTMENU_USERDATA(menu) (* ( (APTR *)(((struct Menu *)menu)+1) ) )
#define GTMENUITEM_USERDATA(menuitem) (* ( (APTR *)(((struct MenuItem *)menuitem
)+1) ) )

/*  Here is an old one for compatibility.  Do not use in new code! */
#define MENU_USERDATA(menuitem) (* ( (APTR *)(menuitem+1) ) )

/*  These return codes can be obtained through the GTMN_ErrorCode tag */
#define GTMENU_TRIMMED  0x00000001      /* Too many menus, items, or subitems,
                                           menu has been trimmed down */
#define GTMENU_INVALID  0x00000002      /* Invalid NewMenu array */
#define GTMENU_NOMEM    0x00000003      /* Out of memory */

/*----------------------------------------------------------------------*/

/*  Tags for toolkit functions: */

#define GT_TagBase           TAG_USER + 0x80000

#define GTVI_NewWindow       GT_TagBase+1  /* Unused */
#define GTVI_NWTags          GT_TagBase+2  /* Unused */

#define GT_Private0          GT_TagBase+3  /* (private) */

#define GTCB_Checked         GT_TagBase+4  /* State of checkbox */

#define GTLV_Top             GT_TagBase+5  /* Top visible one in listview */
```

```
#define GTLV_Labels          GT_TagBase+6  /* List to display in listview */
#define GTLV_ReadOnly        GT_TagBase+7  /* TRUE if listview is to be
                                              read-only */
#define GTLV_ScrollWidth     GT_TagBase+8  /* Width of scrollbar */

#define GTMX_Labels          GT_TagBase+9  /* NULL-terminated array of labels */
#define GTMX_Active          GT_TagBase+10 /* Active one in mx gadget */

#define GTTX_Text            GT_TagBase+11 /* Text to display */
#define GTTX_CopyText        GT_TagBase+12 /* Copy text label instead of
                                              referencing it */

#define GTNM_Number          GT_TagBase+13 /* Number to display */

#define GTCY_Labels          GT_TagBase+14 /* NULL-terminated array of labels */
#define GTCY_Active          GT_TagBase+15 /* The active one in the cycle gad */

#define GTPA_Depth           GT_TagBase+16 /* Number of bitplanes in palette */
#define GTPA_Color           GT_TagBase+17 /* Palette color */
#define GTPA_ColorOffset     GT_TagBase+18 /* First color to use in palette */
#define GTPA_IndicatorWidth  GT_TagBase+19 /* Width of current-color indicator *
 /
#define GTPA_IndicatorHeight GT_TagBase+20 /* Height of current-color indicator
 */

#define GTSC_Top             GT_TagBase+21 /* Top visible in scroller */
#define GTSC_Total           GT_TagBase+22 /* Total in scroller area */
#define GTSC_Visible         GT_TagBase+23 /* Number visible in scroller */
#define GTSC_Overlap         GT_TagBase+24 /* Unused */

/*  GT_TagBase+25 through GT_TagBase+37 are reserved */

#define GTSL_Min             GT_TagBase+38 /* Slider min value */
#define GTSL_Max             GT_TagBase+39 /* Slider max value */
#define GTSL_Level           GT_TagBase+40 /* Slider level */
#define GTSL_MaxLevelLen     GT_TagBase+41 /* Max length of printed level */
#define GTSL_LevelFormat     GT_TagBase+42 /* Format string for level */
#define GTSL_LevelPlace      GT_TagBase+43 /* Where level should be placed */
#define GTSL_DispFunc        GT_TagBase+44 /* Callback for number calculation
                                              before display */

#define GTST_String          GT_TagBase+45 /* String gadget's displayed string *
 /
#define GTST_MaxChars        GT_TagBase+46 /* Max length of string */

#define GTIN_Number          GT_TagBase+47 /* Number in integer gadget */
#define GTIN_MaxChars        GT_TagBase+48 /* Max number of digits */

#define GTMN_TextAttr        GT_TagBase+49 /* MenuItem font TextAttr */
#define GTMN_FrontPen        GT_TagBase+50 /* MenuItem text pen color */

#define GTBB_Recessed        GT_TagBase+51 /* Make BevelBox recessed */

#define GT_VisualInfo        GT_TagBase+52 /* result of VisualInfo call */

#define GTLV_ShowSelected    GT_TagBase+53 /* show selected entry beneath
                listview, set tag data = NULL for display-only, or pointer
                to a string gadget you've created */
#define GTLV_Selected        GT_TagBase+54 /* Set ordinal number of selected
                                              entry in the list */
#define GT_Reserved0         GT_TagBase+55 /* Reserved */
#define GT_Reserved1         GT_TagBase+56 /* Reserved for future use */

#define GTTX_Border          GT_TagBase+57 /* Put a border around
                                              Text-display gadgets */
#define GTNM_Border          GT_TagBase+58 /* Put a border around
```

libraries/gadtools.h

Page 5

```
                                        Number-display gadgets */

#define GTSC_Arrows          GT_TagBase+59 /* Specify size of arrows for
                                              scroller */

#define GTMN_Menu            GT_TagBase+60 /* Pointer to Menu for use by
                                              LayoutMenuItems() */
#define GTMX_Spacing         GT_TagBase+61 /* Added to font height to
                figure spacing between mx choices.  Use this instead
                of LAYOUTA_SPACING for mx gadgets. */

/* New to V37 GadTools.  Ignored by GadTools V36 */
#define GTMN_FullMenu        GT_TagBase+62 /* Asks CreateMenus() to
                validate that this is a complete menu structure */
#define GTMN_SecondaryError  GT_TagBase+63 /* ti_Data is a pointer
                to a ULONG to receive error reports from CreateMenus() */
#define GT_Underscore        GT_TagBase+64 /* ti_Data points to the symbol
                that preceeds the character you'd like to underline in a
                gadget label */

/*  GT_TagBase+65 on up reserved for future use */

/*------------------------------------------------------------------------*/

/*  "NWay" is an old synonym for cycle gadgets */
#define NWAY_KIND       CYCLE_KIND
#define NWAYIDCMP       CYCLEIDCMP
#define GTNW_Labels     GTCY_Labels
#define GTNW_Active     GTCY_Active

/*------------------------------------------------------------------------*/

#endif /* LIBRARIES_GADTOOLS_H */
```

libraries/iffparse.h

Page 1

```
#ifndef IFF_IFFPARSE_H
#define IFF_IFFPARSE_H
/*
**      $Filename: libraries/iffparse.h $
**      $Release: 2.04 $
**      $Revision: 33.1 $
**      $Date: 90/11/20 $
**
**      Structure definitions for the all new good nifty IFF code.
**
**      (C) Copyright 1989,1990 Commodore-Amiga Inc., Stuart Ferguson, and
**      Leo L. Schwab
**              All Rights Reserved
*/

#ifndef EXEC_TYPES_H
#include <exec/types.h>
#endif
#ifndef EXEC_LISTS_H
#include <exec/lists.h>
#endif
#ifndef EXEC_PORTS_H
#include <exec/ports.h>
#endif
#ifndef DEVICES_CLIPBOARD_H
#include <devices/clipboard.h>
#endif

/*
 * Struct associated with an active IFF stream.
 * "iff_Stream" is a value used by the client's read/write/seek functions -
 * it will not be accessed by the library itself and can have any value
 * (could even be a pointer or a BPTR).
 */
struct IFFHandle {
        ULONG   iff_Stream;
        ULONG   iff_Flags;
        LONG    iff_Depth;      /*  Depth of context stack.  */
        /*  There are private fields hiding here.  */
};

/*
 * Bit masks for "iff_Flags" field.
 */
#define IFFF_READ       0L                      /* read mode - default */
#define IFFF_WRITE      1L                      /* write mode */
#define IFFF_RWBITS     (IFFF_READ | IFFF_WRITE)        /* read/write bits */
#define IFFF_FSEEK      (1L<<1)                 /* forward seek only */
#define IFFF_RSEEK      (1L<<2)                 /* random seek */
#define IFFF_RESERVED   0xFFFF0000L             /* Don't touch these bits. */

/*
 * When the library calls your stream handler, you'll be passed a pointer
 * to this structure as the "message packet".
 */
struct IFFStreamCmd {
        LONG    sc_Command;     /*  Operation to be performed (IFFCMD_) */
        APTR    sc_Buf;         /*  Pointer to data buffer              */
        LONG    sc_NBytes;      /*  Number of bytes to be affected      */
};

/*
 * A node associated with a context on the iff_Stack.  Each node
 * represents a chunk, the stack representing the current nesting
 * of chunks in the open IFF file.  Each context node has associated
 * local context items in the (private) LocalItems list.  The ID, type,
 * size and scan values describe the chunk associated with this node.
 */
struct ContextNode {
        struct MinNode  cn_Node;
        LONG            cn_ID;
        LONG            cn_Type;
        LONG            cn_Size;        /* Size of this chunk              */
        LONG            cn_Scan;        /* # of bytes read/written so far */
        /* There are private fields hiding here.  */
};

/*
 * Local context items live in the ContextNode's.  Each class is identified
 * by its lci_Ident code and has a (private) purge vector for when the
 * parent context node is popped.
 */
struct LocalContextItem {
       struct MinNode  lci_Node;
       ULONG           lci_ID,
                       lci_Type,
                       lci_Ident;
        /* There are private fields hiding here.  */
};

/*
 * StoredProperty: a local context item containing the data stored
 * from a previously encountered property chunk.
 */
struct StoredProperty {
        LONG    sp_Size;
        UBYTE   *sp_Data;
};

/*
 * Collection Item: the actual node in the collection list at which
 * client will look.  The next pointers cross context boundaries so
 * that the complete list is accessable.
 */
struct CollectionItem {
        struct CollectionItem   *ci_Next;
        LONG                    ci_Size;
        UBYTE                   *ci_Data;
};

/*
 * Structure returned by OpenClipboard().  You may do CMD_POSTs and such
 * using this structure.  However, once you call OpenIFF(), you may not
 * do any more of your own I/O to the clipboard until you call CloseIFF().
 */
struct ClipboardHandle {
        struct IOClipReq        cbh_Req;
        struct MsgPort          cbh_CBport;
        struct MsgPort          cbh_SatisfyPort;
};

/*
 * IFF return codes.  Most functions return either zero for success or
 * one of these codes.  The exceptions are the read/write functions which
 * return positive values for number of bytes or records read or written,
 * or a negative error code.  Some of these codes are not errors per sae,
 * but valid conditions such as EOF or EOC (End of Chunk).
 */
#define IFFERR_EOF              -1L     /* Reached logical end of file */
#define IFFERR_EOC              -2L     /* About to leave context      */
#define IFFERR_NOSCOPE          -3L     /* No valid scope for property */
#define IFFERR_NOMEM            -4L     /* Internal memory alloc failed*/
```

```
#define IFFERR_READ             -5L     /* Stream read error           */
#define IFFERR_WRITE            -6L     /* Stream write error          */
#define IFFERR_SEEK             -7L     /* Stream seek error           */
#define IFFERR_MANGLED          -8L     /* Data in file is corrupt     */
#define IFFERR_SYNTAX           -9L     /* IFF syntax error            */
#define IFFERR_NOTIFF           -10L    /* Not an IFF file             */
#define IFFERR_NOHOOK           -11L    /* No call-back hook provided  */
#define IFF_RETURN2CLIENT       -12L    /* Client handler normal return*/

#define MAKE_ID(a,b,c,d)        \
        ((ULONG) (a)<<24 | (ULONG) (b)<<16 | (ULONG) (c)<<8 | (ULONG) (d))

/*
 * Universal IFF identifiers.
 */
#define ID_FORM                 MAKE_ID('F','O','R','M')
#define ID_LIST                 MAKE_ID('L','I','S','T')
#define ID_CAT                  MAKE_ID('C','A','T',' ')
#define ID_PROP                 MAKE_ID('P','R','O','P')
#define ID_NULL                 MAKE_ID(' ',' ',' ',' ')

/*
 * Ident codes for universally recognized local context items.
 */
#define IFFLCI_PROP             MAKE_ID('p','r','o','p')
#define IFFLCI_COLLECTION       MAKE_ID('c','o','l','l')
#define IFFLCI_ENTRYHANDLER     MAKE_ID('e','n','h','d')
#define IFFLCI_EXITHANDLER      MAKE_ID('e','x','h','d')

/*
 * Control modes for ParseIFF() function.
 */
#define IFFPARSE_SCAN           0L
#define IFFPARSE_STEP           1L
#define IFFPARSE_RAWSTEP        2L

/*
 * Control modes for StoreLocalItem().
 */
#define IFFSLI_ROOT             1L      /* Store in default context       */
#define IFFSLI_TOP              2L      /* Store in current context       */
#define IFFSLI_PROP             3L      /* Store in topmost FORM or LIST  */

/*
 * "Flag" for writing functions.  If you pass this value in as a size
 * to PushChunk() when writing a file, the parser will figure out the
 * size of the chunk for you.  (Chunk sizes >= 2**31 are forbidden by the
 * IFF specification, so this works.)
 */
#define IFFSIZE_UNKNOWN         -1L

/*
 * Possible call-back command values.  (Using 0 as the value for IFFCMD_INIT
 * was, in retrospect, probably a bad idea.)
 */
#define IFFCMD_INIT     0       /* Prepare the stream for a session    */
#define IFFCMD_CLEANUP  1       /* Terminate stream session            */
#define IFFCMD_READ     2       /* Read bytes from stream              */
#define IFFCMD_WRITE    3       /* Write bytes to stream               */
#define IFFCMD_SEEK     4       /* Seek on stream                      */
#define IFFCMD_ENTRY    5       /* You just entered a new context      */
#define IFFCMD_EXIT     6       /* You're about to leave a context     */
#define IFFCMD_PURGELCI 7       /* Purge a LocalContextItem            */

/* Backward compatibility.  Don't use these in new code.  */
#define IFFSCC_INIT     IFFCMD_INIT
```

libraries/iffparse.h

```
#define IFFSCC_CLEANUP  IFFCMD_CLEANUP
#define IFFSCC_READ     IFFCMD_READ
#define IFFSCC_WRITE    IFFCMD_WRITE
#define IFFSCC_SEEK     IFFCMD_SEEK

#endif
```

libraries/mathffp.h

```
#ifndef LIBRARIES_MATHFFP_H
#define LIBRARIES_MATHFFP_H 1
/*
**      $Filename: libraries/mathffp.h $
**      $Release: 2.04 $
**      $Revision: 36.2 $
**      $Date: 90/05/01 $
**
**      general floating point declarations
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef PI
#define PI        ((float) 3.141592653589793)
#endif
#define TWO_PI    (((float) 2) * PI)
#define PI2       (PI / ((float) 2))
#define PI4       (PI / ((float) 4))
#ifndef E
#define E         ((float) 2.718281828459045)
#endif
#define LOG10     ((float) 2.302585092994046)

#define FPTEN     ((float) 10.0)
#define FPONE     ((float) 1.0)
#define FPHALF    ((float) 0.5)
#define FPZERO    ((float) 0.0)

#define trunc(x)  ((int) (x))
#define round(x)  ((int) ((x) + 0.5))
#define itof(i)   ((float) (i))

#define fabs    SPAbs
#define floor   SPFloor
#define ceil    SPCeil

#define tan     SPTan
#define atan    SPAtan
#define cos     SPCos
#define acos    SPAcos
#define sin     SPSin
#define asin    SPAsin
#define exp     SPExp
#define pow(a,b)        SPPow((b),(a))
#define log     SPLog
#define log10   SPLog10
#define sqrt    SPSqrt

#define sinh    SPSinh
#define cosh    SPCosh
#define tanh    SPTanh


int      SPFix();                            /* Basic math functions */
float    SPFlt();
int      SPCmp();
int      SPTst();
float    SPAbs();
float    SPFloor();
float    SPCeil();
#ifndef abs
float    abs();
#endif
float    SPNeg();
```

libraries/mathffp.h

```
float    SPAdd();
float    SPSub();
float    SPMul();
float    SPDiv();

float    SPAsin(),  SPAcos(),  SPAtan(); /* Transcendental math functions */
float    SPSin(),   SPCos(),   SPTan(),  SPSincos();
float    SPSinh(),  SPCosh(),  SPTanh();
float    SPExp(),   SPLog(),   SPLog10(), SPPow();
float    SPSqrt(),  SPFieee();

float    afp(),    dbf();                /* Math conversion functions */

#endif  /* LIBRARIES_MATHFFP_H */
```

libraries/mathieeedp.h

```
#ifndef LIBRARIES_MATHIEEEDP_H
#define LIBRARIES_MATHIEEEDP_H
/*
**      $Filename: libraries/mathieeedp.h $
**      $Release: 2.04 $
**      $Revision: 1.1 $
**      $Date: 90/07/13 $
**      Include file to use for <math.h>
**
**      (C) Copyright 1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef PI
#define PI      ((double)       3.141592653589793))
#endif

#define TWO_PI  (((double)      2)      * PI)
#define PI2     (PI/((double)2))
#define PI4     (PI/((double)4))

#ifndef E
#define E       ((double)       2.718281828459045)
#endif

#define LOG10   ((double)       2.302585092994046)
#define FPTEN   ((double)       10.0)
#define FPONE   ((double)       1.0)
#define FPHALF  ((double)       0.5)
#define FPZERO  ((double)       0.0)
#define trunc(x)        ((int)  (x))
#define round(x)        ((int)  ((x) + 0.5))
#define itof(i)         ((double) (i))

#define fabs    IEEEDPAbs
#define floor   IEEEDPFloor
#define ceil    IEEEDPCeil

#define tan     IEEEDPTan
#define atan    IEEEDPAtan
#define cos     IEEEDPCos
#define acos    IEEEDPAcos
#define sin     IEEEDPSin
#define asin    IEEEDPAsin
#define exp     IEEEDPExp
#define pow(a,b)        IEEEDPPow((b),(a))
#define log     IEEEDPLog
#define log10   IEEEDPLog10
#define sqrt    IEEEDPSqrt

#define sinh    IEEEDPSinh
#define cosh    IEEEDPCosh
#define tanh    IEEEDPTanh


double  IEEEDPTan(),IEEEDPAtan();
double  IEEEDPCos(),IEEEDPACos();
double  IEEEDPSin(),IEEEDPASin();
double  IEEEDPExp(),IEEEDPLog();
double  IEEEDPSqrt();
double  IEEEDPLog10(),IEEEDPPow();
double  IEEEDPSincos();
double  IEEEDPSinh(),IEEEDPCosh(),IEEEDPTanh();
float   IEEEDPTieee();
double  IEEEDPFieee();

```

libraries/mathieeedp.h

```
int      IEEEDPFix();
int      IEEEDPCmp(),IEEEDPTst();
double   IEEEDPFlt();
double   IEEEDPAbs();
double   IEEEDPNeg();
double   IEEEDPAdd();
double   IEEEDPSub();
double   IEEEDPMul();
double   IEEEDPDiv();
double   IEEEDPFloor();
double   IEEEDPCeil();

#endif   /* LIBRARIES_MATHIEEEDP_H */
```

---

libraries/mathieeesp.h

```
#ifndef LIBRARIES_MATHIEEESP_H
#define LIBRARIES_MATHIEEESP_H
/*
**      $Filename: libraries/mathieeesp.h $
**      $Release: 2.04 $
**      $Revision: 1.1 $
**      $Date: 90/07/13 $
**
**      Include file to use for <math.h>
**
**
**      (C) Copyright 1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef PI
#define PI      ((float)        3.141592653589793))
#endif

#define TWO_PI  (((float)       2)      * PI)
#define PI2     (PI/((float)2))
#define PI4     (PI/((float)4))

#ifndef E
#define E       ((float)        2.718281828459045)
#endif

#define LOG10   ((float)        2.302585092994046)
#define FPTEN   ((float)        10.0)
#define FPONE   ((float)        1.0)
#define FPHALF  ((float)        0.5)
#define FPZERO  ((float)        0.0)
#define trunc(x)        ((int)  (x))
#define round(x)        ((int)  ((x) + 0.5))
#define itof(i)         ((float) (i))

#define fabs    IEEESPAbs
#define floor   IEEESPFloor
#define ceil    IEEESPCeil

#define tan     IEEESPTan
#define atan    IEEESPAtan
#define cos     IEEESPCos
#define acos    IEEESPAcos
#define sin     IEEESPSin
#define asin    IEEESPAsin
#define exp     IEEESPExp
#define pow(a,b)        IEEESPPow((b),(a))
#define log     IEEESPLog
#define log10   IEEESPLog10
#define sqrt    IEEESPSqrt

#define sinh    IEEESPSinh
#define cosh    IEEESPCosh
#define tanh    IEEESPTanh


float   IEEESPTan(),IEEESPAtan();
float   IEEESPCos(),IEEESPACos();
float   IEEESPSin(),IEEESPASin();
float   IEEESPExp(),IEEESPLog();
float   IEEESPSqrt();
float   IEEESPLog10(),IEEESPPow();
float   IEEESPSincos();
float   IEEESPSinh(),IEEESPCosh(),IEEESPTanh();
float   IEEESPTieee();
```

libraries/mathieeesp.h

```
float   IEEESPFieee();

int     IEEESPFix();
int     IEEESPCmp(),IEEESPTst();
float   IEEESPFlt();
float   IEEESPAbs();
float   IEEESPNeg();
float   IEEESPAdd();
float   IEEESPSub();
float   IEEESPMul();
float   IEEESPDiv();
float   IEEESPFloor();
float   IEEESPCeil();

#endif  /* LIBRARIES_MATHIEEESP_H */
```

---

libraries/mathlibrary.h

```
#ifndef LIBRARIES_MATHLIBRARY_H
#define LIBRARIES_MATHLIBRARY_H
/*
**      $Filename: libraries/mathlibrary.h $
**      $Release: 2.04 $
**      $Revision: 1.6 $
**      $Date: 90/07/13 $
**
**      Data structure returned by OpenLibrary of:
**      mathieeedoubbas.library,mathieeedoubtrans.library
**      mathieeesingbas.library,mathieeesingtrans.library
**
**      (C) Copyright 1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_LIBRARIES_H
#include <exec/libraries.h>
#endif

struct MathIEEEBase
{
        struct Library MathIEEEBase_LibNode;
        unsigned char   MathIEEEBase_reserved[18];
        int     (*MathIEEEBase_TaskOpenLib)();
        int     (*MathIEEEBase_TaskCloseLib)();
        /*      This structure may be extended in the future */
};
/*
* Math resources may need to know when a program opens or closes this
* library. The functions TaskOpenLib and TaskCloseLib are called when
* a task opens or closes this library. They are initialized to point to
* local initialization pertaining to 68881 stuff if 68881 resources
* are found. To override the default the vendor must provide appropriate
* hooks in the MathIEEEResource. If specified, these will be called
* when the library initializes.
*/

#endif  /* LIBRARIES_MATHLIBRARY_H */
```

libraries/mathresource.h

```
#ifndef RESOURCES_MATHRESOURCE_H
#define RESOURCES_MATHRESOURCE_H
/*
**      $Filename: libraries/mathresource.h $
**      $Release: 2.04 $
**      $Revision: 1.2 $
**      $Date: 90/07/13 $
**
**      Data structure returned by OpenResource of:
**      "MathIEEE.resource"
**
**
**      (C) Copyright 1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_NODES_H
#include <exec/nodes.h>
#endif

/*
*       The 'Init' entries are only used if the corresponding
*       bit is set in the Flags field.
*
*       So if you are just a 68881, you do not need the Init stuff
*       just make sure you have cleared the Flags field.
*
*       This should allow us to add Extended Precision later.
*
*       For Init users, if you need to be called whenever a task
*       opens this library for use, you need to change the appropriate
*       entries in MathIEEELibrary.
*/

struct MathIEEEResource
{
        struct  Node    MathIEEEResource_Node;
        unsigned short  MathIEEEResource_Flags;
        unsigned short  *MathIEEEResource_BaseAddr; /* ptr to 881 if exists */
        void    (*MathIEEEResource_DblBasInit)();
        void    (*MathIEEEResource_DblTransInit)();
        void    (*MathIEEEResource_SglBasInit)();
        void    (*MathIEEEResource_SglTransInit)();
        void    (*MathIEEEResource_ExtBasInit)();
        void    (*MathIEEEResource_ExtTransInit)();
};

/* definations for MathIEEEResource_FLAGS */
#define MATHIEEERESOURCEF_DBLBAS        (1<<0)
#define MATHIEEERESOURCEF_DBLTRANS      (1<<1)
#define MATHIEEERESOURCEF_SGLBAS        (1<<2)
#define MATHIEEERESOURCEF_SGLTRANS      (1<<3)
#define MATHIEEERESOURCEF_EXTBAS        (1<<4)
#define MATHIEEERESOURCEF_EXTTRANS      (1<<5)

#endif  /* RESOURCES_MATHRESOURCE_H */
```

libraries/translator.h

```
#ifndef LIBRARIES_TRANSLATOR_H
#define LIBRARIES_TRANSLATOR_H

/*
**      $Filename: libraries/translator.h $
**      $Release: 2.04 $
**      $Revision: 36.1 $
**      $Date: 90/12/13 $
**
**      Useful definitions for translator.library
**
**      (C) Copyright 1988, 1989, 1990 Commodore-Amiga Inc. and
**      Joseph Katz/Mark Barton.  All rights reserved.
*/

/*      Translator error return codes   */

#define TR_NotUsed      -1      /* This is an oft used system rc        */
#define TR_NoMem        -2      /* Can't allocate memory                */
#define TR_MakeBad      -4      /* Error in MakeLibrary call            */

#endif  /* LIBRARIES_TRANSLATOR_H */
```

resources/battclock.h

Page 1

```
#ifndef RESOURCES_BATTCLOCK_H
#define RESOURCES_BATTCLOCK_H 1
/*
**      $Filename: resources/battclock.h $
**      $Release: 2.04 $
**      $Revision: 36.4 $
**      $Date: 90/05/01 $
**
**      BattClock resource name strings.
**
**      (C) Copyright 1989,1990 Commodore-Amiga Inc.
**              All Rights Reserved
*/

#define BATTCLOCKNAME   "battclock.resource"

#endif /* RESOURCES_BATTCLOCK_H */
```

resources/battmem.h

Page 1

```
#ifndef RESOURCES_BATTMEM_H
#define RESOURCES_BATTMEM_H 1
/*
**      $Filename: resources/battmem.h $
**      $Release: 2.04 $
**      $Revision: 36.4 $
**      $Date: 90/05/01 $
**
**      BattMem resource name strings.
**
**      (C) Copyright 1989,1990 Commodore-Amiga Inc.
**              All Rights Reserved
*/

#define BATTMEMNAME     "battmem.resource"

#endif /* RESOURCES_BATTMEM_H */
```

resources/battmembitsamiga.h

Page 1

```
#ifndef RESOURCES_BATTMEMBITSAMIGA_H
#define RESOURCES_BATTMEMBITSAMIGA_H 1
/*
**      $Filename: resources/battmembitsamiga.h $
**      $Release: 2.04 $
**      $Revision: 1.1 $
**      $Date: 90/05/25 $
**
**      BattMem Amiga specific bit definitions.
**
**      (C) Copyright 1989,1990 Commodore-Amiga Inc.
**              All Rights Reserved
*/


/*
 * Amiga specific bits in the battery-backedup ram.
 *
 *      Bits 0 to 31, inclusive
 */

/*
 * AMIGA_AMNESIA
 *
 *              The battery-backedup memory has had a memory loss.
 *              This bit is used as a flag that the user should be
 *              notified that all battery-backed bit have been
 *              reset and that some attention is required. Zero
 *              indicates that a memory loss has occured.
 */

#define BATTMEM_AMIGA_AMNESIA_ADDR      0
#define BATTMEM_AMIGA_AMNESIA_LEN       1


/*
 * SCSI_TIMEOUT
 *
 *              adjusts the timeout value for SCSI device selection.  A
 *              value of 0 will produce short timeouts (128 ms) while a
 *              value of 1 produces long timeouts (2 sec).  This is used
 *              for SeaCrate drives (and some Maxtors apparently) that
 *              don't respond to selection until they are fully spun up
 *              and intialised.
 */

#define BATTMEM_SCSI_TIMEOUT_ADDR       1
#define BATTMEM_SCSI_TIMEOUT_LEN        1


/*
 * SCSI_LUNS
 *
 *              Determines if the controller attempts to access logical
 *              units above 0 at any given SCSI address.  This prevents
 *              problems with drives that respond to ALL LUN addresses
 *              (instead of only 0 like they should).  Default value is
 *              0 meaning don't support LUNs.
 */

#define BATTMEM_SCSI_LUNS_ADDR          2
#define BATTMEM_SCSI_LUNS_LEN           1


#endif /* RESOURCES_BATTMEMBITSAMIGA_H */
```

---

resources/battmembitsamix.h

Page 1

```
#ifndef RESOURCES_BATTMEMBITSAMIX_H
#define RESOURCES_BATTMEMBITSAMIX_H 1
/*
**      $Filename: resources/battmembitsamix.h $
**      $Release: 2.04 $
**      $Revision: 1.1 $
**      $Date: 90/05/25 $
**
**      BattMem Amix specific bit definitions.
**
**      (C) Copyright 1989,1990 Commodore-Amiga Inc.
**              All Rights Reserved
*/


/*
 *      See Amix documentation for these bit definitions
 *
 *      Bits 32 to 63, inclusive
 */


#endif /* RESOURCES_BATTMEMBITSAMIX_H */
```

resources/battmembitsshared.h

```
#ifndef RESOURCES_BATTMEMBITSSHARED_H
#define RESOURCES_BATTMEMBITSSHARED_H 1
/*
**      $Filename: resources/battmembitsshared.h $
**      $Release: 2.04 $
**      $Revision: 1.1 $
**      $Date: 90/05/25 $
**
**      BattMem shared specific bit definitions.
**
**      (C) Copyright 1989,1990 Commodore-Amiga Inc.
**              All Rights Reserved
*/


/*
 * Shared bits in the battery-backedup ram.
 *
 *      Bits 64 and above
 */

/*
 * SHARED_AMNESIA
 *
 *              The battery-backedup memory has had a memory loss.
 *              This bit is used as a flag that the user should be
 *              notified that all battery-backed bit have been
 *              reset and that some attention is required. Zero
 *              indicates that a memory loss has occured.
 */

#define BATTMEM_SHARED_AMNESIA_ADDR     64
#define BATTMEM_SHARED_AMNESIA_LEN      1


/*
 * SCSI_HOST_ID
 *
 *              a 3 bit field (0-7) that is stored in complemented form
 *              (this is so that default value of 0 really means 7)
 *              It's used to set the A3000 controllers SCSI ID (on reset)
 */

#define BATTMEM_SCSI_HOST_ID_ADDR       65
#define BATTMEM_SCSI_HOST_ID_LEN        3


/*
 * SCSI_SYNC_XFER
 *
 *              determines if the driver should initiate synchronous
 *              transfer requests or leave it to the drive to send the
 *              first request.  This supports drives that crash or
 *              otherwise get confused when presented with a sync xfer
 *              message.  Default=0=sync xfer not initiated.
 */

#define BATTMEM_SCSI_SYNC_XFER_ADDR     68
#define BATTMEM_SCSI_SYNC_XFER_LEN      1


#endif /* RESOURCES_BATTMEMBITSSHARED_H */
```

resources/cia.h

```
#ifndef DEVICES_CIA_H
#define DEVICES_CIA_H 1
/*
**      $Filename: resources/cia.h $
**      $Release: 2.04 $
**      $Revision: 36.4 $
**      $Date: 91/01/09 $
**
**      Cia resource name strings.
**
**      (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga Inc.
**              All Rights Reserved
*/

#define CIAANAME "ciaa.resource"
#define CIABNAME "ciab.resource"

#endif  /* DEVICES_CIA_H */
```

resources/ciabase.h Page 1

```c
#ifndef RESOURCES_CIA_H
#define RESOURCES_CIA_H
/*
**      $Filename: resources/ciabase.h $
**      $Release: 2.04 $
**      $Revision: 1.2 $
**      $Date: 90/05/16 $
**
**      cia base definitions
**
**      (C) Copyright 1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/


/*
 *      There is no public information in CiaBase
 */


#endif  /* RESOURCES_CIA_H */
```

resources/disk.h Page 1

```c
#ifndef RESOURCES_DISK_H
#define RESOURCES_DISK_H
/*
**      $Filename: resources/disk.h $
**      $Release: 2.04 $
**      $Revision: 27.11 $
**      $Date: 90/11/21 $
**
**      disk.h -- external declarations for the disk resource
**
**      (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_TYPES_H
#include "exec/types.h"
#endif

#ifndef EXEC_LISTS_H
#include "exec/lists.h"
#endif

#ifndef EXEC_PORTS_H
#include "exec/ports.h"
#endif

#ifndef EXEC_INTERRUPTS_H
#include "exec/interrupts.h"
#endif

#ifndef EXEC_LIBRARIES_H
#include "exec/libraries.h"
#endif


/********************************************************************
*
* Resource structures
*
********************************************************************/


struct DiscResourceUnit {
    struct Message dru_Message;
    struct Interrupt dru_DiscBlock;
    struct Interrupt dru_DiscSync;
    struct Interrupt dru_Index;
};

struct DiscResource {
    struct Library              dr_Library;
    struct DiscResourceUnit     *dr_Current;
    UBYTE                       dr_Flags;
    UBYTE                       dr_pad;
    struct Library              *dr_SysLib;
    struct Library              *dr_CiaResource;
    ULONG                       dr_UnitID[4];
    struct List         dr_Waiting;
    struct Interrupt            dr_DiscBlock;
    struct Interrupt            dr_DiscSync;
    struct Interrupt            dr_Index;
    struct Task                 *dr_CurrTask;
};

/* dr_Flags entries */
#define DRB_ALLOC0      0       /* unit zero is allocated */
```

resources/disk.h

```
#define DRB_ALLOC1      1       /* unit one is allocated */
#define DRB_ALLOC2      2       /* unit two is allocated */
#define DRB_ALLOC3      3       /* unit three is allocated */
#define DRB_ACTIVE      7       /* is the disc currently busy? */

#define DRF_ALLOC0      (1<<0)  /* unit zero is allocated */
#define DRF_ALLOC1      (1<<1)  /* unit one is allocated */
#define DRF_ALLOC2      (1<<2)  /* unit two is allocated */
#define DRF_ALLOC3      (1<<3)  /* unit three is allocated */
#define DRF_ACTIVE      (1<<7)  /* is the disc currently busy? */



/**********************************************************************
*
* Hardware Magic
*
**********************************************************************/


#define DSKDMAOFF       0x4000  /* idle command for dsklen register */


/**********************************************************************
*
* Resource specific commands
*
**********************************************************************/

/*
 * DISKNAME is a generic macro to get the name of the resource.
 * This way if the name is ever changed you will pick up the
 *  change automatically.
 */

#define DISKNAME        "disk.resource"


#define DR_ALLOCUNIT    (LIB_BASE - 0*LIB_VECTSIZE)
#define DR_FREEUNIT     (LIB_BASE - 1*LIB_VECTSIZE)
#define DR_GETUNIT      (LIB_BASE - 2*LIB_VECTSIZE)
#define DR_GIVEUNIT     (LIB_BASE - 3*LIB_VECTSIZE)
#define DR_GETUNITID    (LIB_BASE - 4*LIB_VECTSIZE)
#define DR_READUNITID   (LIB_BASE - 5*LIB_VECTSIZE)

#define DR_LASTCOMM     (DR_READUNITID)

/**********************************************************************
*
* drive types
*
**********************************************************************/

#define DRT_AMIGA       (0x00000000)
#define DRT_37422D2S    (0x55555555)
#define DRT_EMPTY       (0xFFFFFFFF)
#define DRT_150RPM      (0xAAAAAAAA)

#endif /* RESOURCES_DISK_H */
```

---

resources/filesysres.h

```
#ifndef RESOURCES_FILESYSRES_H
#define RESOURCES_FILESYSRES_H
/*
**      $Filename: resources/filesysres.h $
**      $Release: 2.04 $
**      $Revision: 36.4 $
**      $Date: 90/05/03 $
**
**      FileSystem.resource description
**
**      (C) Copyright 1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_NODES_H
#include        "exec/nodes.h"
#endif
#ifndef EXEC_LISTS_H
#include        "exec/lists.h"
#endif
#ifndef DOS_DOS_H
#include        "dos/dos.h"
#endif

#define FSRNAME "FileSystem.resource"

struct FileSysResource {
    struct Node fsr_Node;               /* on resource list */
    char   *fsr_Creator;                /* name of creator of this resource */
    struct List fsr_FileSysEntries;     /* list of FileSysEntry structs */
};

struct FileSysEntry {
    struct Node fse_Node;       /* on fsr_FileSysEntries list */
                                /* ln_Name is of creator of this entry */
    ULONG   fse_DosType;        /* DosType of this FileSys */
    ULONG   fse_Version;        /* Version of this FileSys */
    ULONG   fse_PatchFlags;     /* bits set for those of the following that */
                                /*   need to be substituted into a standard */
                                /*   device node for this file system: e.g. */
                                /*   0x180 for substitute SegList & GlobalVec */
    ULONG   fse_Type;           /* device node type: zero */
    CPTR    fse_Task;           /* standard dos "task" field */
    BPTR    fse_Lock;           /* not used for devices: zero */
    BSTR    fse_Handler;        /* filename to loadseg (if SegList is null) */
    ULONG   fse_StackSize;      /* stacksize to use when starting task */
    LONG    fse_Priority;       /* task priority when starting task */
    BPTR    fse_Startup;        /* startup msg: FileSysStartupMsg for disks */
    BPTR    fse_SegList;        /* code to run to start new task */
    BPTR    fse_GlobalVec;      /* BCPL global vector when starting task */
    /* no more entries need exist than those implied by fse_PatchFlags */
};

#endif  /* RESOURCES_FILESYSRES_H */
```

resources/mathresource.h

Page 1

```
#ifndef RESOURCES_MATHRESOURCE_H
#define RESOURCES_MATHRESOURCE_H
/*
**      $Filename: resources/mathresource.h $
**      $Release: 2.04 $
**      $Revision: 1.2 $
**      $Date: 90/07/13 $
**
**      Data structure returned by OpenResource of:
**      "MathIEEE.resource"
**
**
**      (C) Copyright 1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_NODES_H
#include <exec/nodes.h>
#endif

/*
*       The 'Init' entries are only used if the corresponding
*       bit is set in the Flags field.
*
*       So if you are just a 68881, you do not need the Init stuff
*       just make sure you have cleared the Flags field.
*
*       This should allow us to add Extended Precision later.
*
*       For Init users, if you need to be called whenever a task
*       opens this library for use, you need to change the appropriate
*       entries in MathIEEELibrary.
*/

struct MathIEEEResource
{
        struct  Node    MathIEEEResource_Node;
        unsigned short  MathIEEEResource_Flags;
        unsigned short  *MathIEEEResource_BaseAddr; /* ptr to 881 if exists */
        void    (*MathIEEEResource_DblBasInit)();
        void    (*MathIEEEResource_DblTransInit)();
        void    (*MathIEEEResource_SglBasInit)();
        void    (*MathIEEEResource_SglTransInit)();
        void    (*MathIEEEResource_ExtBasInit)();
        void    (*MathIEEEResource_ExtTransInit)();
};

/* definations for MathIEEEResource_FLAGS */
#define MATHIEEERESOURCEF_DBLBAS        (1<<0)
#define MATHIEEERESOURCEF_DBLTRANS      (1<<1)
#define MATHIEEERESOURCEF_SGLBAS        (1<<2)
#define MATHIEEERESOURCEF_SGLTRANS      (1<<3)
#define MATHIEEERESOURCEF_EXTBAS        (1<<4)
#define MATHIEEERESOURCEF_EXTTRANS      (1<<5)

#endif  /* RESOURCES_MATHRESOURCE_H */
```

resources/misc.h

Page 1

```
#ifndef RESOURCES_MISC_H
#define RESOURCES_MISC_H
/*
**      $Filename: resources/misc.h $
**      $Release: 2.04 $
**      $Revision: 36.13 $
**      $Date: 90/05/06 $
**
**      Unit number definitions for "misc.resource"
**
**      (C) Copyright 1985,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
*/

#ifndef EXEC_TYPES_H
#include "exec/types.h"
#endif  /* EXEC_TYPES_H */

#ifndef EXEC_LIBRARIES_H
#include "exec/libraries.h"
#endif  /* EXEC_LIBRARIES_H */

/*
 * Unit number definitions.  Ownership of a resource grants low-level
 * bit access to the hardware registers.  You are still obligated to follow
 * the rules for shared access of the interrupt system (see
 * exec.library/SetIntVector or cia.resource as appropriate).
 */
#define MR_SERIALPORT   0 /* Amiga custom chip serial port registers
                             (SERDAT,SERDATR,SERPER,ADKCON, and interrupts) */
#define MR_SERIALBITS   1 /* Serial control bits (DTR,CTS, etc.) */
#define MR_PARALLELPORT 2 /* The 8 bit parallel data port
                             (CIAAPRA & CIAADDRA only!) */
#define MR_PARALLELBITS 3 /* All other parallel bits & interrupts
                             (BUSY,ACK,etc.) */

/*
 * Library vector offset definitions
 */
#define MR_ALLOCMISCRESOURCE    (LIB_BASE)              /* -6 */
#define MR_FREEMISCRESOURCE     (LIB_BASE-LIB_VECTSIZE) /* -12 */

#define MISCNAME "misc.resource"

#endif  /* RESOURCES_MISC_H */
```

resources/potgo.h

Page 1

```
#ifndef RESOURCES_POTGO_H
#define RESOURCES_POTGO_H
/*
**      $Filename: resources/potgo.h $
**      $Release: 2.04 $
**      $Revision: 36.0 $
**      $Date: 90/04/13 $
**
**      potgo resource name
**
**      (C) Copyright 1986,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
*/
#define  POTGONAME      "potgo.resource"
#endif   /* RESOURCES_POTGO_H */
```

```
#ifndef REXX_ERRORS_H
#define REXX_ERRORS_H
/*
**      $Filename: rexx/errors.h $
**      $Release: 2.04 $
**      $Revision: 1.2 $
**      $Date: 90/07/12 $
**
**      Definitions for ARexx error codes
**
**      (C) Copyright 1987,1988,1989,1990 William S. Hawes
**              All Rights Reserved
*/

#define ERRC_MSG  0                  /*  error code offset           */
#define ERR10_001 (ERRC_MSG+1)       /*  program not found           */
#define ERR10_002 (ERRC_MSG+2)       /*  execution halted            */
#define ERR10_003 (ERRC_MSG+3)       /*  no memory available         */
#define ERR10_004 (ERRC_MSG+4)       /*  invalid character in program*/
#define ERR10_005 (ERRC_MSG+5)       /*  unmatched quote             */
#define ERR10_006 (ERRC_MSG+6)       /*  unterminated comment        */
#define ERR10_007 (ERRC_MSG+7)       /*  clause too long             */
#define ERR10_008 (ERRC_MSG+8)       /*  unrecognized token          */
#define ERR10_009 (ERRC_MSG+9)       /*  symbol or string too long   */

#define ERR10_010 (ERRC_MSG+10)      /*  invalid message packet      */
#define ERR10_011 (ERRC_MSG+11)      /*  command string error        */
#define ERR10_012 (ERRC_MSG+12)      /*  error return from function  */
#define ERR10_013 (ERRC_MSG+13)      /*  host environment not found  */
#define ERR10_014 (ERRC_MSG+14)      /*  required library not found  */
#define ERR10_015 (ERRC_MSG+15)      /*  function not found          */
#define ERR10_016 (ERRC_MSG+16)      /*  no return value             */
#define ERR10_017 (ERRC_MSG+17)      /*  wrong number of arguments   */
#define ERR10_018 (ERRC_MSG+18)      /*  invalid argument to function*/
#define ERR10_019 (ERRC_MSG+19)      /*  invalid PROCEDURE           */

#define ERR10_020 (ERRC_MSG+20)      /*  unexpected THEN/ELSE        */
#define ERR10_021 (ERRC_MSG+21)      /*  unexpected WHEN/OTHERWISE   */
#define ERR10_022 (ERRC_MSG+22)      /*  unexpected LEAVE or ITERATE */
#define ERR10_023 (ERRC_MSG+23)      /*  invalid statement in SELECT */
#define ERR10_024 (ERRC_MSG+24)      /*  missing THEN clauses        */
#define ERR10_025 (ERRC_MSG+25)      /*  missing OTHERWISE           */
#define ERR10_026 (ERRC_MSG+26)      /*  missing or unexpected END   */
#define ERR10_027 (ERRC_MSG+27)      /*  symbol mismatch on END      */
#define ERR10_028 (ERRC_MSG+28)      /*  invalid DO syntax           */
#define ERR10_029 (ERRC_MSG+29)      /*  incomplete DO/IF/SELECT     */

#define ERR10_030 (ERRC_MSG+30)      /*  label not found             */
#define ERR10_031 (ERRC_MSG+31)      /*  symbol expected             */
#define ERR10_032 (ERRC_MSG+32)      /*  string or symbol expected   */
#define ERR10_033 (ERRC_MSG+33)      /*  invalid sub-keyword         */
#define ERR10_034 (ERRC_MSG+34)      /*  required keyword missing    */
#define ERR10_035 (ERRC_MSG+35)      /*  extraneous characters       */
#define ERR10_036 (ERRC_MSG+36)      /*  sub-keyword conflict        */
#define ERR10_037 (ERRC_MSG+37)      /*  invalid template            */
#define ERR10_038 (ERRC_MSG+38)      /*  invalid TRACE request       */
#define ERR10_039 (ERRC_MSG+39)      /*  uninitialized variable      */

#define ERR10_040 (ERRC_MSG+40)      /*  invalid variable name       */
#define ERR10_041 (ERRC_MSG+41)      /*  invalid expression          */
#define ERR10_042 (ERRC_MSG+42)      /*  unbalanced parentheses      */
#define ERR10_043 (ERRC_MSG+43)      /*  nesting level exceeded      */
#define ERR10_044 (ERRC_MSG+44)      /*  invalid expression result   */
#define ERR10_045 (ERRC_MSG+45)      /*  expression required         */
#define ERR10_046 (ERRC_MSG+46)      /*  boolean value not 0 or 1    */
#define ERR10_047 (ERRC_MSG+47)      /*  arithmetic conversion error */
```

```
#define ERR10_048 (ERRC_MSG+48)      /*  invalid operand             */

/*
 * Return Codes for general use
 */
#define RC_OK     0L                 /*  success                     */
#define RC_WARN   5L                 /*  warning only         */
#define RC_ERROR  10L                /*  something's wrong           */
#define RC_FATAL  20L                /*  complete or severe failure  */

#endif
```

```
#ifndef REXX_REXXIO_H
#define REXX_REXXIO_H
/*
**      $Filename: rexx/rexxio.h $
**      $Release: 2.04 $
**      $Revision: 1.2 $
**      $Date: 90/07/12 $
**
**      Header file for ARexx Input/Output related structures
**
**      (C) Copyright 1986,1987,1988,1989,1990 William S. Hawes
**               All Rights Reserved
*/

#ifndef REXX_STORAGE_H
#include "rexx/storage.h"
#endif

#define RXBUFFSZ  204                  /* buffer length         */

/*
 * The IoBuff is a resource node used to maintain the File List.  Nodes
 * are allocated and linked into the list whenever a file is opened.
 */
struct IoBuff {
   struct RexxRsrc iobNode;            /* structure for files/strings   */
   APTR     iobRpt;                    /* read/write pointer            */
   LONG     iobRct;                    /* character count               */
   LONG     iobDFH;                    /* DOS filehandle                */
   APTR     iobLock;                   /* DOS lock                      */
   LONG     iobBct;                    /* buffer length         */
   BYTE     iobArea[RXBUFFSZ];         /* buffer area                   */
   };                                  /* size: 256 bytes               */

/* Access mode definitions                                              */
#define RXIO_EXIST   -1        /* an external filehandle        */
#define RXIO_STRF    0                 /* a "string file"               */
#define RXIO_READ    1                 /* read-only access              */
#define RXIO_WRITE   2                 /* write mode                    */
#define RXIO_APPEND  3                 /* append mode (existing file)   */

/*
 * Offset anchors for SeekF()
 */
#define RXIO_BEGIN   -1L               /* relative to start             */
#define RXIO_CURR    0L        /* relative to current position */
#define RXIO_END     1L        /* relative to end               */

/* The Library List contains just plain resource nodes.         */

#define LLOFFSET(rrp) (rrp->rr_Arg1)   /* "Query" offset                */
#define LLVERS(rrp)   (rrp->rr_Arg2)   /* library version               */

/*
 * The RexxClipNode structure is used to maintain the Clip List.  The value
 * string is stored as an argstring in the rr_Arg1 field.
 */
#define CLVALUE(rrp) ((STRPTR) rrp->rr_Arg1)

/*
 * A message port structure, maintained as a resource node.  The ReplyList
 * holds packets that have been received but haven't been replied.
 */
struct RexxMsgPort {
   struct RexxRsrc rmp_Node;           /* linkage node                  */
   struct MsgPort  rmp_Port;           /* the message port              */
```

```
   struct List     rmp_ReplyList;      /* messages awaiting reply       */
   };

/*
 * DOS Device types
 */
#define DT_DEV     0L                  /* a device                      */
#define DT_DIR     1L                  /* an ASSIGNed directory */
#define DT_VOL     2L                  /* a volume                      */

/*
 * Private DOS packet types
 */
#define ACTION_STACK 2002L             /* stack a line                  */
#define ACTION_QUEUE 2003L             /* queue a line                  */

#endif
```

rexx/rxslib.h Page 1

```
#ifndef REXX_RXSLIB_H
#define REXX_RXSLIB_H
/*
**      $Filename: rexx/rxslib.h $
**      $Release: 2.04 $
**      $Revision: 1.4 $
**      $Date: 90/09/27 $
**
**      The header file for the REXX Systems Library
**
**      (C) Copyright 1987,1988,1989,1990 William S. Hawes
**              All Rights Reserved
*/

#ifndef REXX_STORAGE_H
#include "rexx/storage.h"
#endif

#define RXSNAME  "rexxsyslib.library"
#define RXSDIR   "REXX"
#define RXSTNAME "ARexx"

/* The REXX systems library structure.  This should be considered as    */
/* semi-private and read-only, except for documented exceptions.        */

struct RxsLib {
   struct Library rl_Node;              /* EXEC library node            */
   UBYTE    rl_Flags;                   /* global flags                 */
   UBYTE    rl_Shadow;                  /* shadow flags                 */
   APTR     rl_SysBase;         /* EXEC library base            */
   APTR     rl_DOSBase;         /* DOS library base             */
   APTR     rl_IeeeDPBase;              /* IEEE DP math library base    */
   LONG     rl_SegList;         /* library seglist              */
   LONG     rl_NIL;                     /* global NIL: filehandle       */
   LONG     rl_Chunk;                   /* allocation quantum           */
   LONG     rl_MaxNest;         /* maximum expression nesting   */
   struct NexxStr *rl_NULL;             /* static string: NULL          */
   struct NexxStr *rl_FALSE;            /* static string: FALSE         */
   struct NexxStr *rl_TRUE;             /* static string: TRUE          */
   struct NexxStr *rl_REXX;             /* static string: REXX          */
   struct NexxStr *rl_COMMAND;          /* static string: COMMAND       */
   struct NexxStr *rl_STDIN;            /* static string: STDIN         */
   struct NexxStr *rl_STDOUT;           /* static string: STDOUT */
   struct NexxStr *rl_STDERR;           /* static string: STDERR */
   STRPTR    rl_Version;                /* version string               */

   STRPTR    rl_TaskName;               /* name string for tasks */
   LONG      rl_TaskPri;                /* starting priority            */
   LONG      rl_TaskSeg;                /* startup seglist              */
   LONG      rl_StackSize;              /* stack size                   */
   STRPTR    rl_RexxDir;                /* REXX directory               */
   STRPTR    rl_CTABLE;         /* character attribute table    */
   STRPTR    rl_Notice;         /* copyright notice             */

   struct MsgPort rl_RexxPort;          /* REXX public port             */
   UWORD     rl_ReadLock;               /* lock count                   */
   LONG      rl_TraceFH;                /* global trace console         */
   struct List rl_TaskList;             /* REXX task list               */
   WORD      rl_NumTask;                /* task count                   */
   struct List rl_LibList;              /* Library List header          */
   WORD      rl_NumLib;         /* library count        */
   struct List rl_ClipList;             /* ClipList header              */
   WORD      rl_NumClip;                /* clip node count              */
   struct List rl_MsgList;              /* pending messages             */
   WORD      rl_NumMsg;         /* pending count        */
   struct List rl_PgmList;              /* cached programs              */
```

rexx/rxslib.h Page 2

```
   WORD      rl_NumPgm;         /* program count        */

   UWORD     rl_TraceCnt;               /* usage count for trace console */
   WORD      rl_avail;
   };

/* Global flag bit definitions for RexxMaster                           */
#define RLFB_TRACE RTFB_TRACE           /* interactive tracing?         */
#define RLFB_HALT  RTFB_HALT            /* halt execution?              */
#define RLFB_SUSP  RTFB_SUSP            /* suspend execution?           */
#define RLFB_STOP  6                    /* deny further invocations     */
#define RLFB_CLOSE 7                    /* close the master             */

#define RLFMASK    (1<<RLFB_TRACE) | (1<<RLFB_HALT) | (1<<RLFB_SUSP)

/* Initialization constants                                             */
#define RXSCHUNK   1024         /* allocation quantum           */
#define RXSNEST    32                   /* expression nesting limit     */
#define RXSTPRI    0                    /* task priority        */
#define RXSSTACK   4096         /* stack size                   */

/* Character attribute flag bits used in REXX.                          */
#define CTB_SPACE   0                   /* white space characters       */
#define CTB_DIGIT   1                   /* decimal digits 0-9           */
#define CTB_ALPHA   2                   /* alphabetic characters */
#define CTB_REXXSYM 3                   /* REXX symbol characters       */
#define CTB_REXXOPR 4                   /* REXX operator characters     */
#define CTB_REXXSPC 5                   /* REXX special symbols         */
#define CTB_UPPER   6                   /* UPPERCASE alphabetic         */
#define CTB_LOWER   7                   /* lowercase alphabetic         */

/* Attribute flags                                                      */
#define CTF_SPACE   (1 << CTB_SPACE)
#define CTF_DIGIT   (1 << CTB_DIGIT)
#define CTF_ALPHA   (1 << CTB_ALPHA)
#define CTF_REXXSYM (1 << CTB_REXXSYM)
#define CTF_REXXOPR (1 << CTB_REXXOPR)
#define CTF_REXXSPC (1 << CTB_REXXSPC)
#define CTF_UPPER   (1 << CTB_UPPER)
#define CTF_LOWER   (1 << CTB_LOWER)

#endif
```

```
#ifndef REXX_STORAGE_H
#define REXX_STORAGE_H
/*
**      $Filename: rexx/storage.h $
**      $Release: 2.04 $
**      $Revision: 1.2 $
**      $Date: 90/07/12 $
**
**      Header file to define ARexx data structures.
**
**      (C) Copyright 1986,1987,1988,1989,1990 William S. Hawes
**              All Rights Reserved
*/

#ifndef EXEC_TYPES_H
#include "exec/types.h"
#endif

#ifndef EXEC_NODES_H
#include "exec/nodes.h"
#endif

#ifndef EXEC_LISTS_H
#include "exec/lists.h"
#endif

#ifndef EXEC_PORTS_H
#include "exec/ports.h"
#endif

#ifndef EXEC_LIBRARIES_H
#include "exec/libraries.h"
#endif

/* The NexxStr structure is used to maintain the internal strings in REXX.
 * It includes the buffer area for the string and associated attributes.
 * This is actually a variable-length structure; it is allocated for a
 * specific length string, and the length is never modified thereafter
 * (since it's used for recycling).
 */

struct NexxStr {
   LONG     ns_Ivalue;                 /* integer value         */
   UWORD    ns_Length;                 /* length in bytes (excl null)   */
   UBYTE    ns_Flags;                  /* attribute flags               */
   UBYTE    ns_Hash;                   /* hash code                     */
   BYTE     ns_Buff[8];         /* buffer area for strings       */
   };                                  /* size: 16 bytes (minimum)      */

#define NXADDLEN 9                     /* offset plus null byte */
#define IVALUE(nsPtr) (nsPtr->ns_Ivalue)

/* String attribute flag bit definitions                                */
#define NSB_KEEP      0                /* permanent string?             */
#define NSB_STRING    1                /* string form valid?            */
#define NSB_NOTNUM    2                /* non-numeric?                  */
#define NSB_NUMBER    3                /* a valid number?               */
#define NSB_BINARY    4                /* integer value saved?          */
#define NSB_FLOAT     5                /* floating point format?        */
#define NSB_EXT       6                /* an external string?           */
#define NSB_SOURCE    7                /* part of the program source?   */

/* The flag form of the string attributes                               */
#define NSF_KEEP      (1 << NSB_KEEP  )
#define NSF_STRING    (1 << NSB_STRING)
#define NSF_NOTNUM    (1 << NSB_NOTNUM)
```

```
#define NSF_NUMBER   (1 << NSB_NUMBER)
#define NSF_BINARY   (1 << NSB_BINARY)
#define NSF_FLOAT    (1 << NSB_FLOAT )
#define NSF_EXT      (1 << NSB_EXT   )
#define NSF_SOURCE   (1 << NSB_SOURCE)

/* Combinations of flags                                                 */
#define NSF_INTNUM   (NSF_NUMBER | NSF_BINARY | NSF_STRING)
#define NSF_DPNUM    (NSF_NUMBER | NSF_FLOAT)
#define NSF_ALPHA    (NSF_NOTNUM | NSF_STRING)
#define NSF_OWNED    (NSF_SOURCE | NSF_EXT    | NSF_KEEP)
#define KEEPSTR      (NSF_STRING | NSF_SOURCE | NSF_NOTNUM)
#define KEEPNUM      (NSF_STRING | NSF_SOURCE | NSF_NUMBER | NSF_BINARY)

/* The RexxArg structure is identical to the NexxStr structure, but
 * is allocated from system memory rather than from internal storage.
 * This structure is used for passing arguments to external programs.
 * It is usually passed as an "argstring", a pointer to the string buffer.
 */

struct RexxArg {
   LONG     ra_Size;                   /* total allocated length        */
   UWORD    ra_Length;                 /* length of string              */
   UBYTE    ra_Flags;                  /* attribute flags               */
   UBYTE    ra_Hash;                   /* hash code                     */
   BYTE     ra_Buff[8];         /* buffer area                  */
   };                                  /* size: 16 bytes (minimum)      */

/* The RexxMsg structure is used for all communications with REXX
 * programs.  It is an EXEC message with a parameter block appended.
 */

struct RexxMsg {
   struct Message rm_Node;             /* EXEC message structure        */
   APTR     rm_TaskBlock;              /* global structure (private)    */
   APTR     rm_LibBase;         /* library base (private)       */
   LONG     rm_Action;                 /* command (action) code */
   LONG     rm_Result1;         /* primary result (return code)  */
   LONG     rm_Result2;         /* secondary result             */
   STRPTR   rm_Args[16];               /* argument block (ARG0-ARG15)   */

   struct MsgPort *rm_PassPort;        /* forwarding port               */
   STRPTR   rm_CommAddr;               /* host address (port name)      */
   STRPTR   rm_FileExt;         /* file extension               */
   LONG     rm_Stdin;                  /* input stream (filehandle)     */
   LONG     rm_Stdout;                 /* output stream (filehandle)    */
   LONG     rm_avail;                  /* future expansion              */
   };                                  /* size: 128 bytes               */

/* Field definitions                                                     */
#define ARG0(rmp) (rmp->rm_Args[0])    /* start of argblock             */
#define ARG1(rmp) (rmp->rm_Args[1])    /* first argument                */
#define ARG2(rmp) (rmp->rm_Args[2])    /* second argument               */

#define MAXRMARG  15                   /* maximum arguments             */

/* Command (action) codes for message packets                            */
#define RXCOMM    0x01000000           /* a command-level invocation    */
#define RXFUNC    0x02000000           /* a function call               */
#define RXCLOSE   0x03000000           /* close the REXX server */
#define RXQUERY   0x04000000           /* query for information */
#define RXADDFH   0x07000000           /* add a function host           */
#define RXADDLIB  0x08000000           /* add a function library        */
#define RXREMLIB  0x09000000           /* remove a function library     */
#define RXADDCON  0x0A000000           /* add/update a ClipList string  */
#define RXREMCON  0x0B000000           /* remove a ClipList string      */
```

rexx/storage.h Page 3

```
#define RXTCOPN    0x0C000000          /* open the trace console       */
#define RXTCCLS    0x0D000000          /* close the trace console      */

/* Command modifier flag bits                                          */
#define RXFB_NOIO    16         /* suppress I/O inheritance?    */
#define RXFB_RESULT  17         /* result string expected?      */
#define RXFB_STRING  18         /* program is a "string file"?  */
#define RXFB_TOKEN   19         /* tokenize the command line?   */
#define RXFB_NONRET  20         /* a "no-return" message?       */

/* The flag form of the command modifiers                              */
#define RXFF_NOIO    (1L << RXFB_NOIO  )
#define RXFF_RESULT  (1L << RXFB_RESULT)
#define RXFF_STRING  (1L << RXFB_STRING)
#define RXFF_TOKEN   (1L << RXFB_TOKEN )
#define RXFF_NONRET  (1L << RXFB_NONRET)

#define RXCODEMASK   0xFF000000
#define RXARGMASK    0x0000000F

/* The RexxRsrc structure is used to manage global resources.  Each node
 * has a name string created as a RexxArg structure, and the total size
 * of the node is saved in the "rr_Size" field.  The REXX systems library
 * provides functions to allocate and release resource nodes.  If special
 * deletion operations are required, an offset and base can be provided in
 * "rr_Func" and "rr_Base", respectively.  This "autodelete" function will
 * be called with the base in register A6 and the node in A0.
 */

struct RexxRsrc {
   struct Node rr_Node;
   WORD     rr_Func;                   /* "auto-delete" offset         */
   APTR     rr_Base;                   /* "auto-delete" base           */
   LONG     rr_Size;                   /* total size of node           */
   LONG     rr_Arg1;                   /* available ...        */
   LONG     rr_Arg2;                   /* available ...        */
   };                                  /* size: 32 bytes               */

/* Resource node types                                                 */
#define RRT_ANY      0                 /* any node type ...            */
#define RRT_LIB      1                 /* a function library           */
#define RRT_PORT     2                 /* a public port        */
#define RRT_FILE     3                 /* a file IoBuff        */
#define RRT_HOST     4                 /* a function host              */
#define RRT_CLIP     5                 /* a Clip List node             */

/* The RexxTask structure holds the fields used by REXX to communicate with
 * external processes, including the client task.  It includes the global
 * data structure (and the base environment).  The structure is passed to
 * the newly-created task in its "wake-up" message.
 */

#define GLOBALSZ  200                  /* total size of GlobalData     */

struct RexxTask {
   BYTE     rt_Global[GLOBALSZ];       /* global data structure */
   struct MsgPort rt_MsgPort;          /* global message port          */
   UBYTE    rt_Flags;                  /* task flag bits               */
   BYTE     rt_SigBit;                 /* signal bit                   */

   APTR     rt_ClientID;               /* the client's task ID         */
   APTR     rt_MsgPkt;                 /* the packet being processed   */
   APTR     rt_TaskID;                 /* our task ID                  */
   APTR     rt_RexxPort;               /* the REXX public port         */

   APTR     rt_ErrTrap;         /* Error trap address           */
```

rexx/storage.h Page 4

```
   APTR     rt_StackPtr;               /* stack pointer for traps      */

   struct List rt_Header1;             /* Environment list             */
   struct List rt_Header2;             /* Memory freelist              */
   struct List rt_Header3;             /* Memory allocation list       */
   struct List rt_Header4;             /* Files list                   */
   struct List rt_Header5;             /* Message Ports List           */
   };

/* Definitions for RexxTask flag bits                                  */
#define RTFB_TRACE   0                 /* external trace flag          */
#define RTFB_HALT    1                 /* external halt flag           */
#define RTFB_SUSP    2                 /* suspend task?        */
#define RTFB_TCUSE   3                 /* trace console in use? */
#define RTFB_WAIT    6                 /* waiting for reply?           */
#define RTFB_CLOSE   7                 /* task completed?              */

/* Definitions for memory allocation constants                         */
#define MEMQUANT  16L                  /* quantum of memory space      */
#define MEMMASK   0xFFFFFFF0           /* mask for rounding the size   */

#define MEMQUICK  (1L << 0 )           /* EXEC flags: MEMF_PUBLIC      */
#define MEMCLEAR  (1L << 16)           /* EXEC flags: MEMF_CLEAR       */

/* The SrcNode is a temporary structure used to hold values destined for
 * a segment array.  It is also used to maintain the memory freelist.
 */

struct SrcNode {
   struct SrcNode *sn_Succ;            /* next node                    */
   struct SrcNode *sn_Pred;            /* previous node        */
   APTR     sn_Ptr;                    /* pointer value        */
   LONG     sn_Size;                   /* size of object               */
   };                                  /* size: 16 bytes               */

#endif
```

utility/date.h

```
#ifndef UTILITY_DATE_H
#define UTILITY_DATE_H 1
/*
**      $Filename: utility/date.h $
**      $Release: 2.04 $
**      $Revision: 36.2 $
**      $Date: 91/03/04 $
**
**      Date conversion routines ClockData definition.
**
**      (C) Copyright 1989,1990 Commodore-Amiga Inc.
**              All Rights Reserved
*/

#ifndef EXEC_TYPES_H
#include "exec/types.h"
#endif

struct ClockData
        {
        UWORD sec;
        UWORD min;
        UWORD hour;
        UWORD mday;
        UWORD month;
        UWORD year;
        UWORD wday;
        };

#endif
```

utility/hooks.h

```
#ifndef UTILITY_HOOKS_H
#define UTILITY_HOOKS_H TRUE
/*
**      $Filename: utility/hooks.h $
**      $Release: 2.04 $
**      $Revision: 36.1 $
**      $Date: 90/07/12 $
**
**      callback hooks
**
**      (C) Copyright 1989,1990 Commodore-Amiga Inc.
**              All Rights Reserved
*/

#ifndef EXEC_TYPES_H
#include "exec/types.h"
#endif

#ifndef EXEC_NODES_H
#include "exec/nodes.h"
#endif

/* new standard hook structure */
struct Hook     {
    struct MinNode      h_MinNode;
    ULONG               (*h_Entry)();   /* assembler entry point        */
    ULONG               (*h_SubEntry)();/* often HLL entry point        */
    VOID                *h_Data;        /* owner specific               */
};

/*
 * Hook calling conventions:
 *      A0 - pointer to hook data structure itself
 *      A1 - pointer to parameter structure ("message") typically
 *           beginning with a longword command code, which makes
 *           sense in the context in which the hook is being used.
 *      A2 - Hook specific address data ("object," e.g, GadgetInfo)
 *
 * Control will be passed to the routine h_Entry.  For many
 * High-Level Languages (HLL), this will be an assembly language
 * stub which pushes registers on the stack, does other setup,
 * and then calls the function at h_SubEntry.
 *
 * The C standard receiving code is:
 * CDispatcher( hook, object, message )
 *     struct Hook      *hook;
 *     APTR             object;
 *     APTR             message;
 *
 * NOTE that register natural order differs from this convention
 * for C parameter order, which is A0,A2,A1.
 *
 * The assembly language stub for "vanilla" C parameter conventions
 * could be:

 _hookEntry:
        move.l  a1,-(sp)                ; push message packet pointer
        move.l  a2,-(sp)                ; push object pointer
        move.l  a0,-(sp)                ; push hook pointer
        move.l  h_SubEntry(a0),a0       ; fetch C entry point ...
        jsr     (a0)                    ; ... and call it
        lea     12(sp),sp               ; fix stack
        rts

 * with this function as your interface stub, you can write
 * a Hook setup function as:
```

utility/hooks.h

```

 SetupHook( hook, c_function, userdata )
 struct Hook    *hook;
 ULONG          (*c_function)();
 VOID           *userdata;
 {
        ULONG   (*hookEntry)();

        hook->h_Entry =         hookEntry;
        hook->h_SubEntry =      c_function;
        hook->h_Data =                  userdata;
 }

 * with Lattice C pragmas, you can put the C function in the
 * h_Entry field directly if you declare the function:

ULONG __saveds __asm
CDispatcher(    register __a0 struct Hook       *hook,
                register __a2 VOID              *object,
                register __a1 ULONG             *message );
 *
 ****/

#endif
```

utility/tagitem.h

```
#ifndef UTILITY_TAGITEM_H
#define UTILITY_TAGITEM_H TRUE
/*
**      $Filename: utility/tagitem.h $
**      $Release: 2.04 $
**      $Revision: 36.1 $
**      $Date: 90/07/12 $
**
**      extended specification mechanism
**
**      (C) Copyright 1989,1990 Commodore-Amiga Inc.
**              All Rights Reserved
*/

#ifndef EXEC_TYPES_H
#include <exec/types.h>
#endif

/* ======================================================================= */
/* ==== TagItem ========================================================== */
/* ======================================================================= */
/* This data type may propagate through the system for more general use.
 * In the meantime, it is used as a general mechanism of extensible data
 * arrays for parameter specification and property inquiry (coming soon
 * to a display controller near you).
 *
 * In practice, an array (or chain of arrays) of TagItems is used.
 */

typedef ULONG   Tag;

struct TagItem  {
    Tag         ti_Tag;
    ULONG       ti_Data;
};

/* ---- system tag values ----------------------------- */
#define TAG_DONE   (0L) /* terminates array of TagItems. ti_Data unused */
#define TAG_END TAG_DONE
#define TAG_IGNORE (1L) /* ignore this item, not end of array          */
#define TAG_MORE   (2L) /* ti_Data is pointer to another array of TagItems
                         * note that this tag terminates the current array
                         */
#define TAG_SKIP   (3L) /* skip this and the next ti_Data items        */

/* ---- user tag identification ----------------------- */
#define TAG_USER  (1L<<31)    /* differentiates user tags from system tags*/

/* until further notice, tag bits 16-30 are RESERVED and should be zero.
 * Also, the value (TAG_USER | 0) should never be used as a tag value.
 */

/* ---- Tag filter logic specifiers ---- */
#define TAGFILTER_AND   0       /* exclude everything but filter hits   */
#define TAGFILTER_NOT   1       /* exclude only filter hits             */

#endif
```

workbench/icon.h

```
#ifndef WORKBENCH_ICON_H
#define WORKBENCH_ICON_H
/*
**      $Filename: workbench/icon.h $
**      $Release: 2.04 $
**      $Revision: 36.1 $
**      $Date: 90/10/26 $
**
**      external declarations for icon.library
**
**      (C) Copyright 1985,1986,1987,1988,1989,1990, Commodore-Amiga, Inc.
**      All Rights Reserved
*/

#define ICONNAME        "icon.library"

#endif  /* !WORKBENCH_ICON_H */
```

workbench/startup.h

```
#ifndef WORKBENCH_STARTUP_H
#define WORKBENCH_STARTUP_H
/*
**      $Filename: workbench/startup.h $
**      $Release: 2.04 $
**      $Revision: 36.3 $
**      $Date: 90/07/11 $
**
**      workbench startup definitions
**
**      (C) Copyright 1985,1986,1987,1988,1989,1990, Commodore-Amiga, Inc.
**      All Rights Reserved
*/

#ifndef EXEC_TYPES_H
#include "exec/types.h"
#endif

#ifndef EXEC_PORTS_H
#include "exec/ports.h"
#endif

#ifndef LIBRARIES_DOS_H
#include "libraries/dos.h"
#endif

struct WBStartup {
    struct Message      sm_Message;     /* a standard message structure */
    struct MsgPort *    sm_Process;     /* the process descriptor for you */
    BPTR                sm_Segment;     /* a descriptor for your code */
    LONG                sm_NumArgs;     /* the number of elements in ArgList */
    char *              sm_ToolWindow;  /* description of window */
    struct WBArg *      sm_ArgList;     /* the arguments themselves */
};

struct WBArg {
    BPTR                wa_Lock;        /* a lock descriptor */
    BYTE *              wa_Name;        /* a string relative to that lock */
};

#endif  /* !WORKBENCH_STARTUP_H */
```

```
#ifndef WORKBENCH_WORKBENCH_H
#define WORKBENCH_WORKBENCH_H
/*
**      $Filename: workbench/workbench.h $
**      $Release: 2.04 $
**      $Revision: 36.25 $
**      $Date: 90/11/02 $
**
**      workbench.library general definitions
**
**      (C) Copyright 1985,1986,1987,1988,1989,1990, Commodore-Amiga, Inc.
**      All Rights Reserved
*/

#ifndef EXEC_TYPES_H
#include "exec/types.h"
#endif

#ifndef EXEC_NODES_H
#include "exec/nodes.h"
#endif

#ifndef EXEC_LISTS_H
#include "exec/lists.h"
#endif

#ifndef EXEC_TASKS_H
#include "exec/tasks.h"
#endif

#ifndef INTUITION_INTUITION_H
#include "intuition/intuition.h"
#endif

#define WBDISK          1
#define WBDRAWER        2
#define WBTOOL          3
#define WBPROJECT       4
#define WBGARBAGE       5
#define WBDEVICE        6
#define WBKICK          7
#define WBAPPICON       8

struct OldDrawerData { /* pre V36 definition */
    struct NewWindow    dd_NewWindow;   /* args to open window */
    LONG                dd_CurrentX;    /* current x coordinate of origin */
    LONG                dd_CurrentY;    /* current y coordinate of origin */
};
/* the amount of DrawerData actually written to disk */
#define OLDDRAWERDATAFILESIZE   (sizeof(struct OldDrawerData))

struct DrawerData {
    struct NewWindow    dd_NewWindow;   /* args to open window */
    LONG                dd_CurrentX;    /* current x coordinate of origin */
    LONG                dd_CurrentY;    /* current y coordinate of origin */
    ULONG               dd_Flags;       /* flags for drawer */
    UWORD               dd_ViewModes;   /* view mode for drawer */
};
/* the amount of DrawerData actually written to disk */
#define DRAWERDATAFILESIZE      (sizeof(struct DrawerData))

struct DiskObject {
    UWORD               do_Magic; /* a magic number at the start of the file */
    UWORD               do_Version; /* a version number, so we can change it */
    struct Gadget       do_Gadget;      /* a copy of in core gadget */
    UBYTE               do_Type;
```

```
    char *              do_DefaultTool;
    char **             do_ToolTypes;
    LONG                do_CurrentX;
    LONG                do_CurrentY;
    struct DrawerData * do_DrawerData;
    char *              do_ToolWindow;  /* only applies to tools */
    LONG                do_StackSize;   /* only applies to tools */

};

#define WB_DISKMAGIC    0xe310  /* a magic number, not easily impersonated */
#define WB_DISKVERSION  1       /* our current version number */
#define WB_DISKREVISION 1       /* our current revision number */
/* I only use the lower 8 bits of Gadget.UserData for the revision # */
#define WB_DISKREVISIONMASK     255

struct FreeList {
    WORD                fl_NumFree;
    struct List         fl_MemList;
};

/* each message that comes into the WorkBenchPort must have a type field
** in the preceeding short.  These are the defines for this type
*/

#define MTYPE_PSTD              1       /* a "standard Potion" message */
#define MTYPE_TOOLEXIT          2       /* exit message from our tools */
#define MTYPE_DISKCHANGE        3       /* dos telling us of a disk change */
#define MTYPE_TIMER             4       /* we got a timer tick */
#define MTYPE_CLOSEDOWN         5       /* <unimplemented> */
#define MTYPE_IOPROC            6       /* <unimplemented> */
#define MTYPE_APPWINDOW         7       /* msg from an app window */
#define MTYPE_APPICON           8       /* msg from an app icon */
#define MTYPE_APPMENUITEM       9       /* msg from an app menuitem */
#define MTYPE_COPYEXIT          10      /* exit msg from copy process */
#define MTYPE_ICONPUT           11      /* msg from PutDiskObject in icon.librar
  y */

/* workbench does different complement modes for its gadgets.
** It supports separate images, complement mode, and backfill mode.
** The first two are identical to intuitions GADGIMAGE and GADGHCOMP.
** backfill is similar to GADGHCOMP, but the region outside of the
** image (which normally would be color three when complemented)
** is flood-filled to color zero.
*/
#define GADGBACKFILL    0x0001

/* if an icon does not really live anywhere, set its current position
** to here
*/
#define NO_ICON_POSITION        (0x80000000)

/* workbench now is a library.  this is it's name */
#define WORKBENCH_NAME          "workbench.library"

/* If you find am_Version >= AM_VERSION, you know this structure has
 * at least the fields defined in this version of the include file
 */
#define AM_VERSION      1

struct AppMessage {
    struct Message am_Message;  /* standard message structure */
    UWORD am_Type;              /* message type */
    ULONG am_UserData;          /* application specific */
    ULONG am_ID;                /* application definable ID */
    LONG am_NumArgs;            /* # of elements in arglist */
    struct WBArg *am_ArgList;   /* the arguements themselves */
    UWORD am_Version;           /* will be AM_VERSION */
    UWORD am_Class;             /* message class */
    WORD am_MouseX;             /* mouse x position of event */
    WORD am_MouseY;             /* mouse y position of event */
    ULONG am_Seconds;           /* current system clock time */
    ULONG am_Micros;            /* current system clock time */
    ULONG am_Reserved[8];       /* avoid recompilation */
};

/*
 * The following structures are private.  These are just stub
 * structures for code compatibility...
 */
struct  AppWindow       { void *aw_PRIVATE;  };
struct  AppIcon         { void *ai_PRIVATE;  };
struct          AppMenuItem     { void *ami_PRIVATE; };

#endif  /* !WORKBENCH_WORKBENCH_H */
```

Assembly language include files

This section contains the 68000 assembly language include files from the operating system source code. Whenever the system software requires that a certain structure or constant, it will be defined here. Each subsystem has its own include files.

As with the C language include files, this section is for reference only. Similar include files generally come on disk with whatever assembler you may choose to use with the Amiga. It is important to keep in mind that the include files that come with your assembler may not be an exact match to the files listed here.

WARNING: Not all information in this section should be used in your programs. The include files contain definitions for some structure members and constants that are not supported for use by programs. Many of these are marked as private in the comment field of the include file.

Listed below is a simple example of an assembly language program which uses the header file dos.h.

```
*
* A quick example of using an assembly language include file.  The
* constant "RETURN_FAIL" is not defined in this example, instead the
* value is pulled from the "libraries/dos.i" include file.  This is
* equivalent to:
*
*               moveq   #20,d0
*               rts
*
            INCLUDE "libraries/dos.i"

            moveq   #RETURN_FAIL,D0
            rts
```

devices/audio.i

```
        IFND    DEVICES_AUDIO_I
DEVICES_AUDIO_I SET     1
**
**      $Filename: devices/audio.i $
**      $Release: 2.04 $
**      $Revision: 36.3 $
**      $Date: 90/08/29 $
**
**      audio.device include file
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

        IFND    EXEC_IO_I
        INCLUDE "exec/io.i"
        ENDC

AUDIONAME MACRO
                DC.B    'audio.device',0
        ENDM

ADHARD_CHANNELS         EQU     4

ADALLOC_MINPREC         EQU     -128
ADALLOC_MAXPREC         EQU     127

ADCMD_FREE              EQU     CMD_NONSTD+0
ADCMD_SETPREC           EQU     CMD_NONSTD+1
ADCMD_FINISH            EQU     CMD_NONSTD+2
ADCMD_PERVOL            EQU     CMD_NONSTD+3
ADCMD_LOCK              EQU     CMD_NONSTD+4
ADCMD_WAITCYCLE         EQU     CMD_NONSTD+5
ADCMD_ALLOCATE          EQU     32

ADIOB_PERVOL            EQU     4
ADIOF_PERVOL            EQU     1<<4
ADIOB_SYNCCYCLE         EQU     5
ADIOF_SYNCCYCLE         EQU     1<<5
ADIOB_NOWAIT            EQU     6
ADIOF_NOWAIT            EQU     1<<6
ADIOB_WRITEMESSAGE      EQU     7
ADIOF_WRITEMESSAGE      EQU     1<<7

ADIOERR_NOALLOCATION    EQU     -10
ADIOERR_ALLOCFAILED     EQU     -11
ADIOERR_CHANNELSTOLEN   EQU     -12

    STRUCTURE   IOAudio,IO_SIZE
        WORD    ioa_AllocKey
        APTR    ioa_Data
        ULONG   ioa_Length
        UWORD   ioa_Period
        UWORD   ioa_Volume
        UWORD   ioa_Cycles
        STRUCT  ioa_WriteMsg,MN_SIZE
        LABEL   ioa_SIZEOF

        ENDC    ; DEVICES_AUDIO_I
```

devices/bootblock.i

```
        IFND    DEVICES_BOOTBLOCK_I
DEVICES_BOOTBLOCK_I     SET     1
**
**      $Filename: devices/bootblock.i $
**      $Release: 2.04 $
**      $Revision: 36.6 $
**      $Date: 90/11/05 $
**
**      floppy BootBlock definition
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

        IFND    EXEC_TYPES_I
        INCLUDE "exec/types.i"
        ENDC

 STRUCTURE BB,0
        STRUCT  BB_ID,4                         ; 4 character identifier
        LONG    BB_CHKSUM                       ; boot block checksum (balance)
        LONG    BB_DOSBLOCK                     ; reserved for DOS patch
        LABEL   BB_ENTRY                        ; bootstrap entry point
        LABEL   BB_SIZE

BOOTSECTS       EQU     2                       ; 1K bootstrap

BBID_DOS        macro                           ; something that is bootable
                dc.b    'DOS',0
                endm

BBID_KICK       macro                           ; firmware image disk
                dc.b    'KICK'
                endm


BBNAME_DOS      EQU     $444F5300               ; 'DOS\0'
BBNAME_KICK     EQU     $4B49434B               ; 'KICK'

        ENDC    ; DEVICES_BOOTBLOCK_I
        IFND    DEVICES_CLIPBOARD_I
DEVICES_CLIPBOARD_I  EQU   1
**
**      $Filename: devices/clipboard.i $
**      $Release: 2.04 $
**      $Revision: 36.5 $
**      $Date: 90/11/02 $
**
**      clipboard.device structure definitions
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

        IFND    EXEC_TYPES_I
        INCLUDE "exec/types.i"
        ENDC
        IFND    EXEC_NODES_I
        INCLUDE "exec/nodes.i"
        ENDC
        IFND    EXEC_LISTS_I
        INCLUDE "exec/lists.i"
        ENDC
        IFND    EXEC_PORTS_I
        INCLUDE "exec/ports.i"
        ENDC
        IFND    EXEC_IO_I
        INCLUDE "exec/io.i"
        ENDC

        DEVINIT

        DEVCMD  CBD_POST
        DEVCMD  CBD_CURRENTREADID
        DEVCMD  CBD_CURRENTWRITEID
        DEVCMD  CBD_CHANGEHOOK

CBERR_OBSOLETEID        EQU     1


 STRUCTURE  ClipboardUnitPartial,0
    STRUCT  cu_Node,LN_SIZE     ; list of units
    ULONG   cu_UnitNum          ; unit number for this unit
    ; the remaining unit data is private to the device


 STRUCTURE  IOClipReq,0
    STRUCT  io_Message,MN_SIZE
    APTR    io_Device           ; device node pointer
    APTR    io_Unit             ; unit node pointer (ClipboardUnitPartial)
    UWORD   io_Command          ; device command
    UBYTE   io_Flags            ; including QUICK and SATISFY
    BYTE    io_Error            ; error or warning num
    ULONG   io_Actual           ; number of bytes transferred
    ULONG   io_Length           ; number of bytes requested
    APTR    io_Data             ; either clip stream or post port
    ULONG   io_Offset           ; offset in clip stream
    LONG    io_ClipID           ; ordinal clip identifier
    LABEL   iocr_SIZEOF



PRIMARY_CLIP    EQU     0       ; primary clip unit

 STRUCTURE  SatisfyMsg,0
    STRUCT  sm_Msg,MN_SIZE      ; the length will be 6
```

```
    UWORD   sm_Unit             ; which clip unit this is
    LONG    sm_ClipID           ; the clip identifier of the post
    LABEL   satisfyMsg_SIZEOF

 STRUCTURE  ClipHookMsg,0
    ULONG   chm_Type            ; zero for this structure format
    LONG    chm_ChangeCmd;      ; command that caused this hook invocation:
                                ;   either CMD_UPDATE or CBD_POST
    LONG    chm_ClipID          ; the clip identifier of the new data

        ENDC    ; DEVICES_CLIPBOARD_I
```

devices/console.i

Page 1

```
        IFND    DEVICES_CONSOLE_I
DEVICES_CONSOLE_I       SET     1
**
**      $Filename: devices/console.i $
**      $Release: 2.04 $
**      $Revision: 36.10 $
**      $Date: 90/11/07 $
**
**      Console device command definitions
**
**      (C) Copyright 1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

        IFND    EXEC_TYPES_I
        INCLUDE "exec/types.i"
        ENDC

        IFND    EXEC_IO_I
        INCLUDE "exec/io.i"
        ENDC    ; EXEC_IO_I

******* Console commands *******
        DEVINIT

        DEVCMD  CD_ASKKEYMAP
        DEVCMD  CD_SETKEYMAP
        DEVCMD  CD_ASKDEFAULTKEYMAP
        DEVCMD  CD_SETDEFAULTKEYMAP

******* SGR parameters

SGR_PRIMARY     EQU     0
SGR_BOLD        EQU     1
SGR_ITALIC      EQU     3
SGR_UNDERSCORE  EQU     4
SGR_NEGATIVE    EQU     7

SGR_NORMAL      EQU     22      ; default foreground color, not bold
SGR_NOTITALIC   EQU     23
SGR_NOTUNDERSCORE EQU   24
SGR_POSITIVE    EQU     27

*  these names refer to the ANSI standard, not the implementation
SGR_BLACK       EQU     30
SGR_RED         EQU     31
SGR_GREEN       EQU     32
SGR_YELLOW      EQU     33
SGR_BLUE        EQU     34
SGR_MAGENTA     EQU     35
SGR_CYAN        EQU     36
SGR_WHITE       EQU     37
SGR_DEFAULT     EQU     39

SGR_BLACKBG     EQU     40
SGR_REDBG       EQU     41
SGR_GREENBG     EQU     42
SGR_YELLOWBG    EQU     43
SGR_BLUEBG      EQU     44
SGR_MAGENTABG   EQU     45
SGR_CYANBG      EQU     46
SGR_WHITEBG     EQU     47
SGR_DEFAULTBG   EQU     49

*  these names refer to the implementation, they are the preferred
*  names for use with the Amiga console device.
```

devices/console.i

Page 2

```
SGR_CLR0        EQU     30
SGR_CLR1        EQU     31
SGR_CLR2        EQU     32
SGR_CLR3        EQU     33
SGR_CLR4        EQU     34
SGR_CLR5        EQU     35
SGR_CLR6        EQU     36
SGR_CLR7        EQU     37

SGR_CLR0BG      EQU     40
SGR_CLR1BG      EQU     41
SGR_CLR2BG      EQU     42
SGR_CLR3BG      EQU     43
SGR_CLR4BG      EQU     44
SGR_CLR5BG      EQU     45
SGR_CLR6BG      EQU     46
SGR_CLR7BG      EQU     47


******   DSR parameters

DSR_CPR         EQU     6

******   CTC parameters
CTC_HSETTAB     EQU     0
CTC_HCLRTAB     EQU     2
CTC_HCLRTABSALL EQU     5

******   TBC parameters
TBC_HCLRTAB     EQU     0
TBC_HCLRTABSALL EQU     3

******   SM and RM parameters
M_LNM           EQU     20      ; linefeed newline mode
M_ASM   MACRO
                DC.B    '>1'    ; auto scroll mode
        ENDM
M_AWM   MACRO
                DC.B    '?7'    ; auto wrap mode
        ENDM

        ENDC    ; DEVICES_CONSOLE_I
```

devices/conunit.i

```
	IFND	DEVICES_CONUNIT_I
DEVICES_CONUNIT_I	SET	1
**
**	$Filename: devices/conunit.i $
**	$Release: 2.04 $
**	$Revision: 36.16 $
**	$Date: 90/11/20 $
**
**	Console device unit definitions
**
**	(C) Copyright 1986,1987,1988,1989 Commodore-Amiga, Inc.
**	    All Rights Reserved
**
	IFND	EXEC_TYPES_I
	INCLUDE "exec/types.i"
	ENDC

	IFND	EXEC_PORTS_I
	INCLUDE "exec/ports.i"
	ENDC

	IFND	DEVICES_CONSOLE_I
	INCLUDE "devices/console.i"
	ENDC

	IFND	DEVICES_KEYMAP_I
	INCLUDE "devices/keymap.i"
	ENDC

	IFND	DEVICES_INPUTEVENT_I
	INCLUDE "devices/inputevent.i"
	ENDC

;------ console unit numbers for OpenDevice()
CONU_LIBRARY	EQU	-1	; no unit, just fill in IO_DEVICE field
CONU_STANDARD	EQU	0	; standard unmapped console


;------ New unit numbers for OpenDevice() - (V36)

CONU_CHARMAP	EQU	1	; bind character map to console
CONU_SNIPMAP	EQU	3	; bind character map w/ snip to console

;------ New flag defines for OpenDevice() - (V37)

CONFLAG_DEFAULT			EQU	0
CONFLAG_NODRAW_ON_NEWSIZE	EQU	1


PMB_ASM		EQU	M_LNM+1		; internal storage bit for AS flag
PMB_AWM		EQU	PMB_ASM+1	; internal storage bit for AW flag
MAXTABS		EQU	80


 STRUCTURE	ConUnit,MP_SIZE
    ;------ read only variables
    APTR	cu_Window		; intuition window bound to this unit
    WORD	cu_XCP			; character position
    WORD	cu_YCP
    WORD	cu_XMax			; max character position
    WORD	cu_YMax
    WORD	cu_XRSize		; character raster size
    WORD	cu_YRSize
    WORD	cu_XROrigin		; raster origin
    WORD	cu_YROrigin
    WORD	cu_XRExtant		; raster maxima
```

devices/conunit.i

```
    WORD	cu_YRExtant
    WORD	cu_XMinShrink		; smallest area intact from resize process
    WORD	cu_YMinShrink
    WORD	cu_XCCP			; cursor position
    WORD	cu_YCCP

    ;------ read/write variables (writes must must be protected)
    ;------ storage for AskKeyMap and SetKeyMap
    STRUCT	cu_KeyMapStruct,km_SIZEOF
    ; ---- tab stops
    STRUCT	cu_TabStops,2*MAXTABS ; 0 at start, 0xffff at end of list

    ;------ console rastport attributes
    BYTE	cu_Mask			; these must appear as in RastPort
    BYTE	cu_FgPen		;   |
    BYTE	cu_BgPen		;   |
    BYTE	cu_AOLPen		;   +
    BYTE	cu_DrawMode		; these must appear as in RastPort
    BYTE	cu_Obsolete1		; was cu_AreaPtSz -- not used in V36
    APTR	cu_Obsolete2		; was cu_AreaPtrn -- not used in V36
    STRUCT	cu_Minterms,8		; console minterms
    APTR	cu_Font			;
    UBYTE	cu_AlgoStyle		; these must appear as in RastPort
    UBYTE	cu_TxFlags		;   +
    UWORD	cu_TxHeight		; these must appear as in RastPort
    UWORD	cu_TxWidth		;   |
    UWORD	cu_TxBaseline		;   |
    WORD	cu_TxSpacing		;   +

    ;------ console MODES and RAW EVENTS switches
    STRUCT	cu_Modes,<(PMB_AWM+7)/8>	; one bit per mode
    STRUCT	cu_RawEvents,<(IECLASS_MAX+8)/8>

    ;------ ensure the ConsUnit structure is even
    ALIGNWORD

    LABEL	ConUnit_SIZEOF

	ENDC	; DEVICES_CONUNIT_I
```

devices/gameport.i

```
        IFND    DEVICES_GAMEPORT_I
DEVICES_GAMEPORT_I      SET     1
**
**      $Filename: devices/gameport.i $
**      $Release: 2.04 $
**      $Revision: 36.1 $
**      $Date: 90/11/05 $
**
**      Game Port device command definitions
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

   IFND  EXEC_TYPES_I
   INCLUDE  "exec/types.i"
   ENDC

   IFND  EXEC_IO_I
   INCLUDE  "exec/io.i"
   ENDC


******* GamePort commands *******
   DEVINIT

   DEVCMD   GPD_READEVENT
   DEVCMD   GPD_ASKCTYPE
   DEVCMD   GPD_SETCTYPE
   DEVCMD   GPD_ASKTRIGGER
   DEVCMD   GPD_SETTRIGGER

******* GamePort structures *******

*  gpt_Keys
   BITDEF       GPT,DOWNKEYS,0
   BITDEF       GPT,UPKEYS,1

 STRUCTURE  GamePortTrigger,0
       UWORD    gpt_Keys          ;key transition triggers
       UWORD    gpt_Timeout       ;time trigger (vertical blank units)
       UWORD    gpt_XDelta        ;X distance trigger
       UWORD    gpt_YDelta        ;Y distance trigger
       LABEL    gpt_SIZEOF

******* Controller Types ******
GPCT_ALLOCATED    EQU   -1      ; allocated by another user
GPCT_NOCONTROLLER EQU   0

GPCT_MOUSE        EQU   1
GPCT_RELJOYSTICK  EQU   2
GPCT_ABSJOYSTICK  EQU   3


******* Errors ******
GPDERR_SETCTYPE   EQU   1       ; this controller not valid at this time

        ENDC    ; DEVICES_GAMEPORT_I
```

devices/hardblocks.i

```
        IFND    DEVICES_HARDBLOCKS_I
DEVICES_HARDBLOCKS_I    SET     1
**
**      $Filename: devices/hardblocks.i $
**      $Release: 2.04 $
**      $Revision: 36.2 $
**      $Date: 90/11/07 $
**
**      File System identifier blocks for hard disks
**
**      (C) Copyright 1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

        IFND    EXEC_TYPES_I
        INCLUDE "exec/types.i"
        ENDC


;---------------------------------------------------------------------
;
;       This file describes blocks of data that exist on a hard disk
;       to describe that disk.  They are not generically accessable to
;       the user as they do not appear on any DOS drive.  The blocks
;       are tagged with a unique identifier, checksummed, and linked
;       together.  The root of these blocks is the RigidDiskBlock.
;
;       The RigidDiskBlock must exist on the disk within the first
;       RDB_LOCATION_LIMIT blocks.  This inhibits the use of the zero
;       cylinder in an AmigaDOS partition: although it is strictly
;       possible to store the RigidDiskBlock data in the reserved
;       area of a partition, this practice is discouraged since the
;       reserved blocks of a partition are overwritten by "Format",
;       "Install", "DiskCopy", etc.  The recommended disk layout,
;       then, is to use the first cylinder(s) to store all the drive
;       data specified by these blocks: i.e. partition descriptions,
;       file system load images, drive bad block maps, spare blocks,
;       etc.
;
;       Though only 512 byte blocks are currently supported by the
;       file system, this proposal tries to be forward-looking by
;       making the block size explicit, and by using only the first
;       256 bytes for all blocks but the LoadSeg data.
;
;---------------------------------------------------------------------

;
;   NOTE
;       optional block addresses below contain $ffffffff to indicate
;       a NULL address
;
 STRUCTURE      RigidDiskBlock,0
    ULONG   rdb_ID                  ; 4 character identifier
    ULONG   rdb_SummedLongs         ; size of this checksummed structure
    LONG    rdb_ChkSum              ; block checksum (longword sum to zero)
    ULONG   rdb_HostID              ; SCSI Target ID of host
    ULONG   rdb_BlockBytes          ; size of disk blocks
    ULONG   rdb_Flags               ; see below for defines
    ; block list heads
    ULONG   rdb_BadBlockList        ; optional bad block list
    ULONG   rdb_PartitionList       ; optional first partition block
    ULONG   rdb_FileSysHeaderList ; optional fule system header block
    ULONG   rdb_DriveInit           ; optional drive-specific init code
                                    ; DriveInit(lun,rdb,ior): "C" stk & d0/a0/a1
    STRUCT  rdb_Reserved1,6*4       ; set to $ffffffff
    ; physical drive characteristics
    ULONG   rdb_Cylinders       ; number of drive cylinders
    ULONG   rdb_Sectors         ; sectors per track
    ULONG   rdb_Heads           ; number of drive heads
    ULONG   rdb_Interleave      ; interleave
    ULONG   rdb_Park            ; landing zone cylinder
    STRUCT  rdb_Reserved2,3*4
    ULONG   rdb_WritePreComp    ; starting cylinder: write precompensation
    ULONG   rdb_ReducedWrite    ; starting cylinder: reduced write current
    ULONG   rdb_StepRate        ; drive step rate
    STRUCT  rdb_Reserved3,5*4
    ; logical drive characteristics
    ULONG   rdb_RDBBlocksLo     ; low block of range reserved for hardblocks
    ULONG   rdb_RDBBlocksHi     ; high block of range for these hardblocks
    ULONG   rdb_LoCylinder      ; low cylinder of partitionable disk area
    ULONG   rdb_HiCylinder      ; high cylinder of partitionable data area
    ULONG   rdb_CylBlocks       ; number of blocks available per cylinder
    ULONG   rdb_AutoParkSeconds ; zero for no auto park
    STRUCT  rdb_Reserved4,2*4
    ; drive identification
    STRUCT  rdb_DiskVendor,8
    STRUCT  rdb_DiskProduct,16
    STRUCT  rdb_DiskRevision,4
    STRUCT  rdb_ControllerVendor,8
    STRUCT  rdb_ControllerProduct,16
    STRUCT  rdb_ControllerRevision,4
    STRUCT  rdb_Reserved5,10*4

    LABEL   RigidDiskBlock_SIZEOF

IDNAME_RIGIDDISK        EQU     (('R'<<24)!('D'<<16)!('S'<<8)!('K'))

RDB_LOCATION_LIMIT      EQU     16

    BITDEF  RDBF,LAST,0         ; no disks exist to be configured after
                                ;   this one on this controller
    BITDEF  RDBF,LASTLUN,1      ; no LUNs exist to be configured greater
                                ;   than this one at this SCSI Target ID
    BITDEF  RDBF,LASTTID,2      ; no Target IDs exist to be configured
                                ;   greater than this one on this SCSI bus
    BITDEF  RDBF,NORESELECT,3   ; don't bother trying to perform reselection
                                ;   when talking to this drive
    BITDEF  RDBF,DISKID,4       ; rdb_Disk... identification valid
    BITDEF  RDBF,CTRLRID,5      ; rdb_Controller... identification valid


;-----------------------------------------------------------------------
 STRUCTURE      BadBlockEntry,0
    ULONG   bbe_BadBlock        ; block number of bad block
    ULONG   bbe_GoodBlock       ; block number of replacement block
    LABEL   BadBlockEntry_SIZEOF

 STRUCTURE      BadBlockBlock,0
    ULONG   bbb_ID              ; 4 character identifier
    ULONG   bbb_SummedLongs     ; size of this checksummed structure
    LONG    bbb_ChkSum          ; block checksum (longword sum to zero)
    ULONG   bbb_HostID          ; SCSI Target ID of host
    ULONG   bbb_Next            ; block number of the next BadBlockBlock
    ULONG   bbb_Reserved
    STRUCT  bbb_BlockPairs,61*BadBlockEntry_SIZEOF ; bad block entry pairs
    ; note 61 assumes 512 byte blocks
    ; there is no BadBlockBlock_SIZEOF: try rdb_BlockBytes

IDNAME_BADBLOCK         EQU     (('B'<<24)!('A'<<16)!('D'<<8)!('B'))

;-----------------------------------------------------------------------
 STRUCTURE      PartitionBlock,0
```

```
    ULONG   pb_ID               ; 4 character identifier
    ULONG   pb_SummedLongs      ; size of this checksummed structure
    LONG    pb_ChkSum           ; block checksum (longword sum to zero)
    ULONG   pb_HostID           ; SCSI Target ID of host
    ULONG   pb_Next             ; block number of the next PartitionBlock
    ULONG   pb_Flags            ; see below for defines
    STRUCT  pb_Reserved1,2*4
    ULONG   pb_DevFlags         ; preferred flags for OpenDevice
    STRUCT  pb_DriveName,32     ; preferred DOS device name: BSTR form
                                ; (not used if this name is in use)
    STRUCT  pb_Reserved2,15*4   ; filler to 32 longwords
    STRUCT  pb_Environment,17*4 ; environment vector for this partition
    STRUCT  pb_EReserved,15*4   ; reserved for future environment vector
    LABEL   PartitionBlock_SIZEOF

IDNAME_PARTITION        EQU     (('P'<<24)!('A'<<16)!('R'<<8)!('T'))

    BITDEF  PBF,BOOTABLE,0      ; this partition is intended to be bootable
                                ;   (expected directories and files exist)
    BITDEF  PBF,NOMOUNT,1       ; do not mount this partition (e.g. manually
                                ;   mounted, but space reserved here)

;-----------------------------------------------------------------------
 STRUCTURE      FileSysHeaderBlock,0
    ULONG   fhb_ID              ; 4 character identifier
    ULONG   fhb_SummedLongs     ; size of this checksummed structure
    LONG    fhb_ChkSum          ; block checksum (longword sum to zero)
    ULONG   fhb_HostID          ; SCSI Target ID of host
    ULONG   fhb_Next            ; block number of the next FileSysHeaderBlock
    ULONG   fhb_Flags           ; see below for defines
    STRUCT  fhb_Reserved1,2*4
    ULONG   fhb_DosType         ; file system description: match this with
                                ; partition environment's DE_DOSTYPE entry
    ULONG   fhb_Version         ; release version of this code
    ULONG   fhb_PatchFlags      ; bits set for those of the following that
                                ;   need to be substituted into a standard
                                ;   device node for this file system: e.g.
                                ;   $180 to substitute SegList & GlobalVec
    ULONG   fhb_Type            ; device node type: zero
    ULONG   fhb_Task            ; standard dos "task" field: zero
    ULONG   fhb_Lock            ; not used for devices: zero
    ULONG   fhb_Handler         ; filename to loadseg: zero placeholder
    ULONG   fhb_StackSize       ; stacksize to use when starting task
    LONG    fhb_Priority        ; task priority when starting task
    LONG    fhb_Startup         ; startup msg: zero placeholder
    LONG    fhb_SegListBlocks   ; first of linked list of LoadSegBlocks:
                                ;   note that this entry requires some
                                ;   processing before substitution
    LONG    fhb_GlobalVec       ; BCPL global vector when starting task
    STRUCT  fhb_Reserved2,23*4  ; (those reserved by PatchFlags)
    STRUCT  fhb_Reserved3,21*4
    LABEL   FileSysHeader_SIZEOF

IDNAME_FILESYSHEADER    EQU     (('F'<<24)!('S'<<16)!('H'<<8)!('D'))

;-----------------------------------------------------------------------
 STRUCTURE      LoadSegBlock,0
    ULONG   lsb_ID              ; 4 character identifier
    ULONG   lsb_SummedLongs     ; size of this checksummed structure
    LONG    lsb_ChkSum          ; block checksum (longword sum to zero)
    ULONG   lsb_HostID          ; SCSI Target ID of host
    ULONG   lsb_Next            ; block number of the next FileSysBlock
    STRUCT  lsb_LoadData,123*4  ; data for "loadseg"
    ; note 123 assumes 512 byte blocks
    ; there is no LoadSegBlock_SIZEOF: try rdb_BlockBytes

```

devices/hardblocks.i Page 4

```
IDNAME_LOADSEG          EQU      (('L'<<24)!('S'<<16)!('E'<<8)!('G'))

        ENDC
```

devices/input.i Page 1

```
        IFND    DEVICES_INPUT_I
DEVICES_INPUT_I SET     1
**
**      $Filename: devices/input.i $
**      $Release: 2.04 $
**      $Revision: 36.0 $
**      $Date: 90/05/01 $
**
**      input device command definitions
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

   IFND     EXEC_IO_I
   INCLUDE     "exec/io.i"
   ENDC

   DEVINIT

   DEVCMD       IND_ADDHANDLER
   DEVCMD       IND_REMHANDLER
   DEVCMD       IND_WRITEEVENT
   DEVCMD       IND_SETTHRESH
   DEVCMD       IND_SETPERIOD
   DEVCMD       IND_SETMPORT
   DEVCMD       IND_SETMTYPE
   DEVCMD       IND_SETMTRIG

        ENDC    ; DEVICES_INPUT_I
        IFND    DEVICES_INPUTEVENT_I
DEVICES_INPUTEVENT_I    SET     1
**
**      $Filename: devices/inputevent.i $
**      $Release: 2.04 $
**      $Revision: 36.6 $
**      $Date: 91/01/22 $
**
**      input event definitions
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

        IFND     DEVICES_TIMER_I
        INCLUDE  "devices/timer.i"
        ENDC

;------ constants ---------------------------------------------------

;   --- InputEvent.ie_Class ---
; A NOP input event
IECLASS_NULL                    EQU     $00
; A raw keycode from the keyboard device
IECLASS_RAWKEY                  EQU     $01
; A raw mouse report from the game port device
IECLASS_RAWMOUSE                EQU     $02
; A private console event
IECLASS_EVENT                   EQU     $03
; A Pointer Position report
IECLASS_POINTERPOS              EQU     $04
; A timer event
IECLASS_TIMER                   EQU     $06
; select button pressed down over a Gadget (address in ie_EventAddress)
IECLASS_GADGETDOWN              EQU     $07
; select button released over the same Gadget (address in ie_EventAddress)
IECLASS_GADGETUP                EQU     $08
; some Requester activity has taken place.  See Codes REQCLEAR and REQSET
IECLASS_REQUESTER               EQU     $09
; this is a Menu Number transmission (Menu number is in ie_Code)
IECLASS_MENULIST                EQU     $0A
; User has selected the active Window's Close Gadget
IECLASS_CLOSEWINDOW             EQU     $0B
; this Window has a new size
IECLASS_SIZEWINDOW              EQU     $0C
; the Window pointed to by ie_EventAddress needs to be refreshed
IECLASS_REFRESHWINDOW           EQU     $0D
; new preferences are available
IECLASS_NEWPREFS                EQU     $0E
; the disk has been removed
IECLASS_DISKREMOVED             EQU     $0F
; the disk has been inserted
IECLASS_DISKINSERTED            EQU     $10
; the window is about to be been made active
IECLASS_ACTIVEWINDOW            EQU     $11
; the window is about to be made inactive
IECLASS_INACTIVEWINDOW          EQU     $12
; extended-function pointer position report (V36)
IECLASS_NEWPOINTERPOS           EQU     $13
; Help key report during Menu session (V36)
IECLASS_MENUHELP                EQU     $14
; the Window has been modified with move, size, zoom, or change (V36)
IECLASS_CHANGEWINDOW            EQU     $15

; the last class
IECLASS_MAX                     EQU     $15
```

```


;   --- InputEvent.ie_SubClass ---
;  IECLASS_NEWPOINTERPOS
;       like IECLASS_POINTERPOS
IESUBCLASS_COMPATIBLE           EQU     $00
;       ie_EventAddress points to struct IEPointerPixel
IESUBCLASS_PIXEL                EQU     $01
;       ie_EventAddress points to struct IEPointerTablet
IESUBCLASS_TABLET               EQU     $02

; pointed to by ie_EventAddress for IECLASS_NEWPOINTERPOS,
; and IESUBCLASS_PIXEL.
;
; You specify a screen and pixel coordinates in that screen
; at which you'd like the mouse to be positioned.
; Intuition will try to oblige, but there will be restrictions
; to positioning the pointer over offscreen pixels.
;
; IEQUALIFIER_RELATIVEMOUSE is supported for IESUBCLASS_PIXEL.

 STRUCTURE      IEPointerPixel,0
    APTR    iepp_Screen         ; pointer to an open screen
    LABEL   iepp_Position       ; pixel coordinates in iepp_Screen
    WORD    iepp_PositionX
    WORD    iepp_PositionY
    LABEL   IEPointerPixel_SIZEOF

; pointed to by ie_EventAddress for IECLASS_NEWPOINTERPOS,
; and IESUBCLASS_TABLET.
;
; You specify a range of values and a value within the range
; independently for each of X and Y (the minimum value of
; the ranges is always normalized to 0).
;
; Intuition will position the mouse proportionally within its
; natural mouse position rectangle limits.
;
; IEQUALIFIER_RELATIVEMOUSE is not supported for IESUBCLASS_TABLET.

 STRUCTURE      IEPointerTablet,0
    LABEL   iept_Range          ; 0 is min, these are max
    UWORD   iept_RangeX
    UWORD   iept_RangeY
    LABEL   iept_Value          ; between 0 and iept_Range
    UWORD   iept_ValueX
    UWORD   iept_ValueY
    WORD    iept_Pressure       ; -128 to 127 (unused, set to 0)
    LABEL   IEPointerTablet_SIZEOF


;   --- InputEvent.ie_Code ---
;   IECLASS_RAWKEY
IECODE_UP_PREFIX                EQU     $80
IECODEB_UP_PREFIX               EQU     7
IECODE_KEY_CODE_FIRST           EQU     $00
IECODE_KEY_CODE_LAST            EQU     $77
IECODE_COMM_CODE_FIRST          EQU     $78
IECODE_COMM_CODE_LAST           EQU     $7F

;   IECLASS_ANSI
IECODE_C0_FIRST                 EQU     $00
IECODE_C0_LAST                  EQU     $1F
IECODE_ASCII_FIRST              EQU     $20
IECODE_ASCII_LAST               EQU     $7E
IECODE_ASCII_DEL                EQU     $7F
```

devices/inputevent.i Page 3

```
IECODE_C1_FIRST                 EQU     $80
IECODE_C1_LAST                  EQU     $9F
IECODE_LATIN1_FIRST             EQU     $A0
IECODE_LATIN1_LAST              EQU     $FF

;   IECLASS_RAWMOUSE
IECODE_LBUTTON                  EQU     $68     ; also uses IECODE_UP_PREFIX
IECODE_RBUTTON                  EQU     $69     ;
IECODE_MBUTTON                  EQU     $6A     ;
IECODE_NOBUTTON                 EQU     $FF

;   IECLASS_EVENT (V36)
IECODE_NEWACTIVE                EQU     $01     ; new active input window
IECODE_NEWSIZE                  EQU     $02     ; resize of specified window
IECODE_REFRESH                  EQU     $03     ; refresh of specified window

;   IECLASS_REQUESTER Codes
;       broadcast when the first Requester (not subsequent ones) opens in
;       the Window
IECODE_REQSET                   EQU     $01
;       broadcast when the last Requester clears out of the Window
IECODE_REQCLEAR                 EQU     $00


*   --- InputEvent.ie_Qualifier ---
IEQUALIFIER_LSHIFT              EQU     $0001
IEQUALIFIER_RSHIFT              EQU     $0002
IEQUALIFIER_CAPSLOCK            EQU     $0004
IEQUALIFIER_CONTROL             EQU     $0008
IEQUALIFIER_LALT                EQU     $0010
IEQUALIFIER_RALT                EQU     $0020
IEQUALIFIER_LCOMMAND            EQU     $0040
IEQUALIFIER_RCOMMAND            EQU     $0080
IEQUALIFIER_NUMERICPAD          EQU     $0100
IEQUALIFIER_REPEAT              EQU     $0200
IEQUALIFIER_INTERRUPT           EQU     $0400
IEQUALIFIER_MULTIBROADCAST      EQU     $0800
IEQUALIFIER_MIDBUTTON           EQU     $1000
IEQUALIFIER_RBUTTON             EQU     $2000
IEQUALIFIER_LEFTBUTTON          EQU     $4000
IEQUALIFIER_RELATIVEMOUSE       EQU     $8000

IEQUALIFIERB_LSHIFT             EQU     0
IEQUALIFIERB_RSHIFT             EQU     1
IEQUALIFIERB_CAPSLOCK           EQU     2
IEQUALIFIERB_CONTROL            EQU     3
IEQUALIFIERB_LALT               EQU     4
IEQUALIFIERB_RALT               EQU     5
IEQUALIFIERB_LCOMMAND           EQU     6
IEQUALIFIERB_RCOMMAND           EQU     7
IEQUALIFIERB_NUMERICPAD         EQU     8
IEQUALIFIERB_REPEAT             EQU     9
IEQUALIFIERB_INTERRUPT          EQU     10
IEQUALIFIERB_MULTIBROADCAST     EQU     11
IEQUALIFIERB_MIDBUTTON          EQU     12
IEQUALIFIERB_RBUTTON            EQU     13
IEQUALIFIERB_LEFTBUTTON         EQU     14
IEQUALIFIERB_RELATIVEMOUSE      EQU     15


;------ InputEvent ----------------------------------------------------

 STRUCTURE      InputEvent,0
    APTR    ie_NextEvent        ; the chronologically next event
    UBYTE   ie_Class            ; the input event class
    UBYTE   ie_SubClass         ; optional subclass of the class
```

devices/inputevent.i Page 4

```
    UWORD   ie_Code             ; the input event code
    UWORD   ie_Qualifier        ; qualifiers in effect for the event
    LABEL   ie_EventAddress     ; the event address
    LABEL   ie_X                ; the pointer position for the event
    UBYTE   ie_Prev1DownCode    ; previous down keys for dead key translation
    UBYTE   ie_Prev1DownQual
    LABEL   ie_Y
    UBYTE   ie_Prev2DownCode
    UBYTE   ie_Prev2DownQual
    STRUCT  ie_TimeStamp,TV_SIZE ; the system tick at the event
    LABEL   ie_SIZEOF

        ENDC    ; DEVICES_INPUTEVENT_I
```

devices/keyboard.i

```
        IFND    DEVICES_KEYBOARD_I
DEVICES_KEYBOARD_I      SET     1
**
**      $Filename: devices/keyboard.i $
**      $Release: 2.04 $
**      $Revision: 36.0 $
**      $Date: 90/05/01 $
**
**      Keyboard device command definitions
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

   IFND     EXEC_IO_I
   INCLUDE      "exec/io.i"
   ENDC

   DEVINIT

   DEVCMD       KBD_READEVENT
   DEVCMD       KBD_READMATRIX
   DEVCMD       KBD_ADDRESETHANDLER
   DEVCMD       KBD_REMRESETHANDLER
   DEVCMD       KBD_RESETHANDLERDONE

        ENDC    ; DEVICES_KEYBOARD_I
```

---

devices/keymap.i

```
        IFND    DEVICES_KEYMAP_I
DEVICES_KEYMAP_I        SET     1
**
**      $Filename: devices/keymap.i $
**      $Release: 2.04 $
**      $Revision: 36.3 $
**      $Date: 90/04/13 $
**
**      key map definitions for keymap.resource, keymap.library, and
**      console.device
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

       IFND     EXEC_NODES_I
       INCLUDE  "exec/nodes.i"
       ENDC
       IFND     EXEC_LISTS_I
       INCLUDE  "exec/lists.i"
       ENDC

 STRUCTURE  KeyMap,0
   APTR  km_LoKeyMapTypes
   APTR  km_LoKeyMap
   APTR  km_LoCapsable
   APTR  km_LoRepeatable
   APTR  km_HiKeyMapTypes
   APTR  km_HiKeyMap
   APTR  km_HiCapsable
   APTR  km_HiRepeatable
   LABEL km_SIZEOF

 STRUCTURE      KeyMapNode,0
    STRUCT  kn_Node,LN_SIZE     ; including name of keymap
    STRUCT  kn_KeyMap,km_SIZEOF
    LABEL   kn_SIZEOF

;------ the structure of keymap.resource
 STRUCTURE      KeyMapResource,0
    STRUCT  kr_Node,LN_SIZE
    STRUCT  kr_List,LH_SIZE     ; a list of KeyMapNodes
    LABEL   kr_SIZEOF


KCB_NOP     EQU   7
KCF_NOP     EQU   $80

KC_NOQUAL   EQU   0
KC_VANILLA  EQU   7            ; note that SHIFT+ALT+CTRL is VANILLA
KCB_SHIFT   EQU   0
KCF_SHIFT   EQU   $01
KCB_ALT     EQU   1
KCF_ALT     EQU   $02
KCB_CONTROL EQU   2
KCF_CONTROL EQU   $04
KCB_DOWNUP  EQU   3
KCF_DOWNUP  EQU   $08
KCB_DEAD    EQU   5              ; may be dead or modified by dead key:
KCF_DEAD    EQU   $20            ;   use dead prefix bytes

KCB_STRING  EQU   6
KCF_STRING  EQU   $40

;------ Dead Prefix Bytes
DPB_MOD EQU     0
```

devices/keymap.i

```
DPF_MOD EQU     $01
DPB_DEAD        EQU     3
DPF_DEAD        EQU     $08

DP_2DINDEXMASK  EQU     $0F     ; mask for index for 1st of two dead keys
DP_2DFACSHIFT   EQU     4       ; shift for factor for 1st of two dead keys

        ENDC    ; DEVICES_KEYMAP_I
```

devices/narrator.i

```
        IFND DEVICES_NARRATOR_I
DEVICES_NARRATOR_I      SET 1
**
**      $Filename: devices/narrator.i $
**      $Release: 2.04 $
**      $Revision: 1.7 $
**      $Date: 91/03/12 $
**
**      V37 Narrator device ASM include file
**
**      Copyright 1990, 1991 Joseph Katz/Mark Barton.
**      All rights reserved.
**
**      This include file (narrator.i) may be freely distributed
**      as long as the above copyright notice remains intact.
**

        IFND    EXEC_IO_I
        INCLUDE "exec/io.i"
        ENDC


*               ;------ Default values, user parms, and general constants

DEFPITCH        EQU     110             ;DEFAULT PITCH
DEFRATE EQU     150             ;DEFAULT RATE
DEFVOL          EQU     64              ;DEFAULT VOLUME (FULL)
DEFFREQ EQU     22200           ;DEFAULT SAMPLING FREQUENCY
NATURALF0       EQU     0               ;NATURAL F0 CONTOURS
ROBOTICF0       EQU     1               ;MONOTONE PITCH
MANUALF0        EQU     2               ;MANUAL SETTING OF PITCH
MALE            EQU     0               ;MALE SPEAKER
FEMALE          EQU     1               ;FEMALE SPEAKER
DEFSEX          EQU     MALE            ;DEFAULT SEX
DEFMODE EQU     NATURALF0       ;DEFAULT MODE
DEFARTIC        EQU     100             ;DEFAULT ARTICULATION 100%
DEFCENTRAL      EQU     0               ;DEFAULT PERCENTAGE OF CENTRALIZATION=0
DEFF0PERT       EQU     0               ;DEFAULT F0 PERTURBATION
DEFF0ENTHUS     EQU     32              ;DEFAULT F0 ENTHUSIASM (in 32nds)
DEFPRIORITY     EQU     100             ;DEFAULT SPEAKING PRIORITY


*               ;------ Parameter bounds

MINRATE EQU      40             ;MINIMUM SPEAKING RATE
MAXRATE EQU     400             ;MAXIMUM SPEAKING RATE
MINPITCH        EQU      65             ;MINIMUM PITCH
MAXPITCH        EQU     320             ;MAXIMUM PITCH
MINFREQ EQU    5000             ;MINIMUM SAMPLING FREQUENCY
MAXFREQ EQU   28000             ;MAXIMUM SAMPLING FREQUENCY
MINVOL          EQU       0             ;MINIMUM VOLUME
MAXVOL          EQU      64             ;MAXIMUM VOLUME
MINCENT EQU       0             ;MINIMUM DEGREE OF CENTRALIZATION
MAXCENT EQU     100             ;MAXIMUM DEGREE OF CENTRALIZATION

*               ;------ Driver error codes

ND_NotUsed      EQU      -1             ;
ND_NoMem        EQU      -2             ;Can't allocate memory
ND_NoAudLib     EQU      -3             ;Can't open audio device
ND_MakeBad      EQU      -4             ;Error in MakeLibrary call
ND_UnitErr      EQU      -5             ;Unit other than 0
ND_CantAlloc    EQU      -6             ;Can't allocate the audio channel
ND_Unimpl       EQU      -7             ;Unimplemented command
ND_NoWrite      EQU      -8             ;Read for mouth shape without write
ND_Expunged     EQU      -9             ;Can't open, deferred expunge bit set
```

```
ND_PhonErr       EQU     -20                  ;Phoneme code spelling error
ND_RateErr       EQU     -21                  ;Rate out of bounds
ND_PitchErr      EQU     -22                  ;Pitch out of bounds
ND_SexErr        EQU     -23                  ;Sex not valid
ND_ModeErr       EQU     -24                  ;Mode not valid
ND_FreqErr       EQU     -25                  ;Sampling freq out of bounds
ND_VolErr        EQU     -26                  ;Volume out of bounds
ND_DCentErr      EQU     -27                  ;Degree of centralization out of bounds
ND_CentPhonErr   EQU     -28                  ;Invalid central phon


*                ;------ Bit/field definitions of "flags" field of IORB.

NDB_NEWIORB      EQU     0                    ;Use new IORB flag
NDB_WORDSYNC     EQU     1                    ;Generate word sync messages
NDB_SYLSYNC      EQU     2                    ;Generate syllable sync messages

NDF_NEWIORB      EQU     (1<<NDB_NEWIORB)
NDF_WORDSYNC     EQU     (1<<NDB_WORDSYNC)
NDF_SYLSYNC      EQU     (1<<NDB_SYLSYNC)


*                ;------ Write IORequest block

 STRUCTURE NDI,IOSTD_SIZE
        UWORD    NDI_RATE                     ;Speaking rate in words/minute
        UWORD    NDI_PITCH                    ;Baseline pitch in Hertz
        UWORD    NDI_MODE                     ;F0 mode
        UWORD    NDI_SEX                      ;Speaker sex
        APTR     NDI_CHMASKS                  ;Pointer to audio channel masks
        UWORD    NDI_NUMMASKS                 ;Size of channel masks array
        UWORD    NDI_VOLUME                   ;Channel volume
        UWORD    NDI_SAMPFREQ                 ;Sampling frequency
        UBYTE    NDI_MOUTHS                   ;Generate mouths? (Boolean value)
        UBYTE    NDI_CHANMASK                 ;Actual channel mask used (internal use)
        UBYTE    NDI_NUMCHAN                  ;Number of channels used (internal use)
        UBYTE    NDI_FLAGS                    ;New feature flags
        UBYTE    NDI_F0ENTHUSIASM             ;F0 excursion factor
        BYTE     NDI_F0PERTURB                ;Amount of F0 perturbation
        BYTE     NDI_F1ADJ                    ;F1 adjustment in M-15% steps
        BYTE     NDI_F2ADJ                    ;F2 adjustment in M-15% steps
        BYTE     NDI_F3ADJ                    ;F3 adjustment in M-15% steps
        BYTE     NDI_A1ADJ                    ;A1 adjustment in decibels
        BYTE     NDI_A2ADJ                    ;A2 adjustment in decibels
        BYTE     NDI_A3ADJ                    ;A3 adjustment in decibels
        UBYTE    NDI_ARTICULATE               ;Transition time multiplier
        UBYTE    NDI_CENTRALIZE               ;Degree of vowel centralization
        APTR     NDI_CENTPHON                 ;Ptr to ASCII central phon code
        BYTE     NDI_AVBIAS                   ;AV bias
        BYTE     NDI_AFBIAS                   ;AF bias
        BYTE     NDI_PRIORITY                 ;Priority while speaking
        BYTE     NDI_PAD1                     ;For alignment
        LABEL    NDI_SIZE                     ;Size of Narrator IORequest block


*                ;------ Mouth read IORB

 STRUCTURE MRB,NDI_SIZE
        UBYTE    MRB_WIDTH                    ;Mouth width
        UBYTE    MRB_HEIGHT                   ;Mouth height
        UBYTE    MRB_SHAPE                    ;Compressed shape (height/width)
        UBYTE    MRB_SYNC                     ;Sync events
        LABEL    MRB_SIZE


        ENDC     ; DEVICES_NARRATOR_I
```

devices/parallel.i Page 1

```
    IFND     DEVICES_PARALLEL_I
DEVICES_PARALLEL_I SET 1
**
**      $Filename: devices/parallel.i $
**      $Release: 2.04 $
**      $Revision: 34.9 $
**      $Date: 89/05/25 $
**
**      external declarations for the parallel device
**
**      (C) Copyright 1985,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**
    IFND     EXEC_IO_I
    include "exec/io.i"
    ENDC    !EXEC_IO_I

*--------------------------------------------------------------------
*
* Driver error definitions
*
*--------------------------------------------------------------------

ParErr_DevBusy          EQU     1
ParErr_BufTooBig                EQU     2
ParErr_InvParam         EQU     3
ParErr_LineErr          EQU     4
ParErr_NotOpen          EQU     5
ParErr_PortReset        EQU     6
ParErr_InitErr          EQU     7

*--------------------------------------------------------------------
*
* Useful constants
*
*--------------------------------------------------------------------
*
PDCMD_QUERY         EQU     CMD_NONSTD
PDCMD_SETPARAMS     EQU     CMD_NONSTD+1
Par_DEVFINISH       EQU     10          ; number of device comands
*
*--------------------------------------------------------------------
*
* Driver Specific Commands
*
*--------------------------------------------------------------------

PARALLELNAME:   MACRO
                dc.b    'parallel.device',0
                ds.w    0
                ENDM

        BITDEF  PAR,SHARED,5        ; PARFLAGS non-exclusive access
        BITDEF  PAR,SLOWMODE,4      ;    "     slow mode selected bit
        BITDEF  PAR,FASTMODE,3      ;    "     fast mode selected bit
        BITDEF  PAR,RAD_BOOGIE,3    ;    "     for backward compatibility
        BITDEF  PAR,ACKMODE,2       ;    "     ACK handshaking selected bit
        BITDEF  PAR,EOFMODE,1       ;    "     EOF mode enabled bit
        BITDEF  IOPAR,QUEUED,6      ; IO_FLAGS rqst-queued bit
        BITDEF  IOPAR,ABORT,5       ;    "     rqst-aborted bit
        BITDEF  IOPAR,ACTIVE,4      ;    "     rqst-qued-or-current bit
        BITDEF  IOPT,RWDIR,3        ; IO_STATUS read=0,write=1
        BITDEF  IOPT,PARSEL,2       ;    "     printer selected on the A1000
                                    ; printer selected & serial "Ring Indicator" o
  n
                                    ; the A500/A2000.  Be careful when making cabl
```

devices/parallel.i Page 2

```
  es.
        BITDEF  IOPT,PAPEROUT,1     ;    "     paper out
        BITDEF  IOPT,PARBUSY,0      ;    "     printer in busy toggle
;Note: Previous versions of this include file had bits 0 and 2 swapped
*
***********************************************************************

 STRUCTURE PTERMARRAY,0
        ULONG   PTERMARRAY_0
        ULONG   PTERMARRAY_1
        LABEL   PTERMARRAY_SIZE

*****************************************************************
*  CAUTION !!!  IF YOU ACCESS the parallel.device, you MUST (!!!!) use an
*  IOEXTPAR-sized structure or you may overlay innocent memory, okay ?!
*****************************************************************

 STRUCTURE IOEXTPAR,IOSTD_SIZE

*     STRUCT   MsgNode
*   0   APTR     Succ
*   4   APTR     Pred
*   8   UBYTE    Type
*   9   UBYTE    Pri
*   A   APTR     Name
*   E   APTR     ReplyPort
*  12   UWORD    MNLength
*     STRUCT   IOExt
*  14   APTR     IO_DEVICE
*  18   APTR     IO_UNIT
*  1C   UWORD    IO_COMMAND
*  1E   UBYTE    IO_FLAGS
*  1F   UBYTE    IO_ERROR
*     STRUCT   IOStdExt
*  20   ULONG    IO_ACTUAL
*  24   ULONG    IO_LENGTH
*  28   APTR     IO_DATA
*  2C   ULONG    IO_OFFSET
*

*
*  30
        ULONG   IO_PEXTFLAGS    ; (not used) flag extension area
        UBYTE   IO_PARSTATUS    ; device status (see bit defs above)
        UBYTE   IO_PARFLAGS     ; see PARFLAGS bit definitions above
        STRUCT  IO_PTERMARRAY,PTERMARRAY_SIZE ; termination char array
        LABEL   IOEXTPar_SIZE

***************************************************************************

    ENDC    !DEVICES_PARALLEL_I
```

```
   IFND  DEVICES_PRINTER_I
DEVICES_PRINTER_I EQU   1
**
**      $Filename: devices/printer.i $
**      $Release: 2.04 $
**      $Revision: 1.7 $
**      $Date: 90/07/26 $
**
**      printer.device structure definitions
**
**      (C) Copyright 1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**

   IFND  EXEC_TYPES_I
   INCLUDE  "exec/types.i"
   ENDC

   IFND  EXEC_NODES_I
   INCLUDE  "exec/nodes.i"
   ENDC

   IFND  EXEC_LISTS_I
   INCLUDE  "exec/lists.i"
   ENDC

   IFND  EXEC_PORTS_I
   INCLUDE  "exec/ports.i"
   ENDC

   IFND  EXEC_IO_I
   INCLUDE  "exec/io.i"
   ENDC

   DEVINIT

   DEVCMD   PRD_RAWWRITE
   DEVCMD   PRD_PRTCOMMAND
   DEVCMD   PRD_DUMPRPORT
   DEVCMD   PRD_QUERY

;****** printer definitions
aRIS     EQU    0 ; ESCc  reset              ISO
aRIN     EQU    1 ; ESC#1 initialize                 +++
aIND     EQU    2 ; ESCD  lf                         ISO
aNEL     EQU    3 ; ESCE  return,lf                  ISO
aRI      EQU    4 ; ESCM  reverse lf                 ISO

aSGR0    EQU    5 ; ESC[0m normal char set           ISO
aSGR3    EQU    6 ; ESC[3m italics on                ISO
aSGR23   EQU    7 ; ESC[23m italics off      ISO
aSGR4    EQU    8 ; ESC[4m underline on      ISO
aSGR24   EQU    9 ; ESC[24m underline off            ISO
aSGR1    EQU   10 ; ESC[1m boldface on               ISO
aSGR22   EQU   11 ; ESC[22m boldface off             ISO
aSFC     EQU   12 ; SGR30-39   set foreground color  ISO
aSBC     EQU   13 ; SGR40-49   set background color  ISO

aSHORP0  EQU   14 ; ESC[0w normal pitch      DEC
aSHORP2  EQU   15 ; ESC[2w elite on                  DEC
aSHORP1  EQU   16 ; ESC[1w elite off                 DEC
aSHORP4  EQU   17 ; ESC[4w condensed fine on         DEC
aSHORP3  EQU   18 ; ESC[3w condensed off             DEC
aSHORP6  EQU   19 ; ESC[6w enlarged on               DEC
aSHORP5  EQU   20 ; ESC[5w enlarged off      DEC

```

```
aDEN6    EQU   21 ; ESC[6"z shadow print on          DEC (sort of)
aDEN5    EQU   22 ; ESC[5"z shadow print off         DEC
aDEN4    EQU   23 ; ESC[4"z doublestrike on          DEC
aDEN3    EQU   24 ; ESC[3"z doublestrike off         DEC
aDEN2    EQU   25 ; ESC[2"z  NLQ on                  DEC
aDEN1    EQU   26 ; ESC[1"z  NLQ off                 DEC

aSUS2    EQU   27 ; ESC[2v superscript on            +++
aSUS1    EQU   28 ; ESC[1v superscript off           +++
aSUS4    EQU   29 ; ESC[4v subscript on      +++
aSUS3    EQU   30 ; ESC[3v subscript off             +++
aSUS0    EQU   31 ; ESC[0v normalize the line        +++
aPLU     EQU   32 ; ESCL  partial line up            ISO
aPLD     EQU   33 ; ESCK  partial line down          ISO

aFNT0    EQU   34 ; ESC(B US char set       or Typeface  0 (default)
aFNT1    EQU   35 ; ESC(R French char set   or Typeface  1
aFNT2    EQU   36 ; ESC(K German char set   or Typeface  2
aFNT3    EQU   37 ; ESC(A UK char set       or Typeface  3
aFNT4    EQU   38 ; ESC(E Danish I char set or Typeface  4
aFNT5    EQU   39 ; ESC(H Sweden char set   or Typeface  5
aFNT6    EQU   40 ; ESC(Y Italian char set  or Typeface  6
aFNT7    EQU   41 ; ESC(Z Spanish char set  or Typeface  7
aFNT8    EQU   42 ; ESC(J Japanese char set or Typeface  8
aFNT9    EQU   43 ; ESC(6 Norweign char set or Typeface  9
aFNT10   EQU   44 ; ESC(C Danish II char set or Typeface 10

;       Suggested typefaces are:
;
;        0 - default typeface.
;        1 - Line Printer or equiv.
;        2 - Pica or equiv.
;        3 - Elite or equiv.
;        4 - Helvetica or equiv.
;        5 - Times Roman or equiv.
;        6 - Gothic or equiv.
;        7 - Script or equiv.
;        8 - Prestige or equiv.
;        9 - Caslon or equiv.
;       10 - Orator or equiv.
;

aPROP2   EQU   45 ; ESC[2p  proportional on          +++
aPROP1   EQU   46 ; ESC[1p  proportional off         +++
aPROP0   EQU   47 ; ESC[0p  proportional clear       +++
aTSS     EQU   48 ; ESC[n E set proportional offset  ISO
aJFY5    EQU   49 ; ESC[5 F auto left justify        ISO
aJFY7    EQU   50 ; ESC[7 F auto right justiy        ISO
aJFY6    EQU   51 ; ESC[6 F auto full justify        ISO
aJFY0    EQU   52 ; ESC[0 F auto justify off         ISO
aJFY2    EQU   53 ; ESC[2 F  word space(auto center) ISO (special)
aJFY3    EQU   54 ; ESC[3 F letter space (justify)   ISO (special)

aVERP0   EQU   55 ; ESC[0z  1/8" line spacing        +++
aVERP1   EQU   56 ; ESC[1z  1/6" line spacing        +++
aSLPP    EQU   57 ; ESC[nt  set form length n        DEC
aPERF    EQU   58 ; ESC[nq  perf skip n (n>0)        +++
aPERF0   EQU   59 ; ESC[0q  perf skip off            +++

aLMS     EQU   60 ; ESC#9  Left margin set           +++
aRMS     EQU   61 ; ESC#0  Right margin set          +++
aTMS     EQU   62 ; ESC#8  Top margin set            +++
aBMS     EQU   63 ; ESC#2  Bottom marg set           +++
aSTBM    EQU   64 ; ESC[Pn1;Pn2r  T&B margins        DEC
aSLRM    EQU   65 ; ESC[Pn1;Pn2s  L&R margin         DEC
aCAM     EQU   66 ; ESC#3  Clear margins             +++

aHTS     EQU   67 ; ESCH     Set horiz tab               ISO
aVTS     EQU   68 ; ESCJ     Set vertical tabs           ISO
aTBC0    EQU   69 ; ESC[0g   Clr horiz tab               ISO
aTBC3    EQU   70 ; ESC[3g   Clear all h tab             ISO
aTBC1    EQU   71 ; ESC[1g   Clr vertical tabs           ISO
aTBC4    EQU   72 ; ESC[4g   Clr all v tabs              ISO
aTBCALL  EQU   73 ; ESC#4    Clr all h & v tabs          +++
aTBSALL  EQU   74 ; ESC#5    Set default tabs            +++
aEXTEND  EQU   75 ; ESC[Pn"x extended commands           +++

aRAW     EQU   76 ; ESC[Pn"r Next 'Pn' chars are raw  +++


  STRUCTURE IOPrtCmdReq,IO_SIZE
     UWORD     io_PrtCommand  ; printer command
     UBYTE     io_Parm0     ; first command parameter
     UBYTE     io_Parm1     ; second command parameter
     UBYTE     io_Parm2     ; third command parameter
     UBYTE     io_Parm3     ; fourth command parameter
     LABEL     iopcr_SIZEOF

  STRUCTURE  IODRPReq,IO_SIZE
     APTR      io_RastPort     ; raster port
     APTR      io_ColorMap     ; color map
     ULONG     io_Modes        ; graphics viewport modes
     UWORD     io_SrcX         ; source x origin
     UWORD     io_SrcY         ; source y origin
     UWORD     io_SrcWidth     ; source x width
     UWORD     io_SrcHeight    ; source x height
     LONG      io_DestCols     ; destination x width
     LONG      io_DestRows     ; destination y height
     UWORD     io_Special      ; option flags
     LABEL     iodrpr_SIZEOF

SPECIAL_MILCOLS          EQU     $0001   ; DestCols specified in 1/1000"
SPECIAL_MILROWS          EQU     $0002   ; DestRows specified in 1/1000"
SPECIAL_FULLCOLS         EQU     $0004   ; make DestCols maximum possible
SPECIAL_FULLROWS         EQU     $0008   ; make DestRows maximum possible
SPECIAL_FRACCOLS         EQU     $0010   ; DestCols is fraction of FULLCOLS
SPECIAL_FRACROWS         EQU     $0020   ; DestRows is fraction of FULLROWS
SPECIAL_CENTER           EQU     $0040   ; center image on paper
SPECIAL_ASPECT           EQU     $0080   ; ensure correct aspect ratio
SPECIAL_DENSITY1         EQU     $0100   ; lowest resolution (dpi)
SPECIAL_DENSITY2         EQU     $0200   ; next res
SPECIAL_DENSITY3         EQU     $0300   ; next res
SPECIAL_DENSITY4         EQU     $0400   ; next res
SPECIAL_DENSITY5         EQU     $0500   ; next res
SPECIAL_DENSITY6         EQU     $0600   ; next res
SPECIAL_DENSITY7         EQU     $0700   ; highest res
SPECIAL_NOFORMFEED       EQU     $0800   ; don't eject paper after gfx prints
SPECIAL_TRUSTME          EQU     $1000   ; don't reset on gfx prints
;
;       Compute print size, set 'io_DestCols' and 'io_DestRows' in the calling
;       program's 'IODRPReq' structure and exit, don't print.  This allows the
;       calling program to see what the final print size would be in printer
;       pixels.  Note that it modifies the 'io_DestCols' and 'io_DestRows'
;       fields of your 'IODRPReq' structure.  Also, set the print density and
;       update the 'MaxXDots', 'MaxYDots', 'XDotsInch', and 'YDotsInch' fields
;       of the 'PrinterExtendedData' structure.
;
SPECIAL_NOPRINT          EQU     $2000   ; see above

PDERR_NOERR              EQU     0       ; clean exit, no errors
PDERR_CANCEL             EQU     1       ; user cancelled print
PDERR_NOTGRAPHICS        EQU     2       ; printer cannot output graphics
```

```
PDERR_INVERTHAM         EQU     3       ; OBSOLETE
PDERR_BADDIMENSION      EQU     4       ; print dimensions illegal
PDERR_DIMENSIONOVFLOW   EQU     5       ; OBSOLETE
PDERR_INTERNALMEMORY    EQU     6       ; no memory for internal variables
PDERR_BUFFERMEMORY      EQU     7       ; no memory for print buffer
;
;       Note : this is an internal error that can be returned from the render
;       function to the printer device.  It is NEVER returned to the user.
;       If the printer device sees this error it converts it 'PDERR_NOERR'
;       and exits gracefully.  Refer to the document on
;       'How to Write a Graphics Printer Driver' for more info.
;
PDERR_TOOKCONTROL       EQU     8       ; I took control in case 0 of render

; internal use
SPECIAL_DENSITYMASK     EQU $0700       ;  masks out density values
SPECIAL_DIMENSIONSMASK  EQU SPECIAL_MILCOLS!SPECIAL_MILROWS!SPECIAL_FULLCOLS!SPE
  CIAL_FULLROWS!SPECIAL_FRACCOLS!SPECIAL_FRACROWS!SPECIAL_ASPECT

   ENDC
```

```
    IFND  DEVICES_PRTBASE_I
DEVICES_PRTBASE_I EQU   1
**
**      $Filename: devices/prtbase.i $
**      $Release: 2.04 $
**      $Revision: 1.9 $
**      $Date: 90/07/26 $
**
**      printer.device base structure definitions
**
**      (C) Copyright 1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

    IFND  EXEC_TYPES_I
    INCLUDE  "exec/types.i"
    ENDC
    IFND  EXEC_NODES_I
    INCLUDE  "exec/nodes.i"
    ENDC
    IFND  EXEC_LISTS_I
    INCLUDE  "exec/lists.i"
    ENDC
    IFND  EXEC_PORTS_I
    INCLUDE  "exec/ports.i"
    ENDC
    IFND  EXEC_LIBRARIES_I
    INCLUDE  "exec/libraries.i"
    ENDC
    IFND  EXEC_TASKS_I
    INCLUDE  "exec/tasks.i"
    ENDC

    IFND  DEVICES_PARALLEL_I
    INCLUDE  "devices/parallel.i"
    ENDC
    IFND  DEVICES_SERIAL_I
    INCLUDE  "devices/serial.i"
    ENDC
    IFND  DEVICES_TIMER_I
    INCLUDE  "devices/timer.i"
    ENDC
    IFND  LIBRARIES_DOSEXTENS_I
    INCLUDE  "libraries/dosextens.i"
    ENDC
    IFND  INTUITION_INTUITION_I
    INCLUDE  "intuition/intuition.i"
    ENDC


 STRUCTURE  DeviceData,LIB_SIZE
    APTR dd_Segment             ; A0 when initialized
    APTR dd_ExecBase            ; A6 for exec
    APTR dd_CmdVectors          ; command table for device commands
    APTR dd_CmdBytes            ; bytes describing which command queue
    UWORD   dd_NumCommands      ; the number of commands supported
    LABEL   dd_SIZEOF


*------
*------ device driver private variables ------------------------------
*------
du_Flags EQU    LN_PRI          ; various unit flags

;------ IO_FLAGS
    BITDEF  IO,QUEUED,4         ; command is queued to be performed
```

```
    BITDEF  IO,CURRENT,5        ; command is being performed
    BITDEF  IO,SERVICING,6      ; command is being actively performed
    BITDEF  IO,DONE,7           ; command is done

;------ du_Flags
    BITDEF  DU,STOPPED,0        ; commands are not to be performed


*------ Constants -------------------------------------------------------
P_PRIORITY      EQU     0
P_OLDSTKSIZE    EQU     $0800   ; stack size for child task (OBSOLETE)
P_STKSIZE       EQU     $1000   ; stack size for child task
P_BUFSIZE       EQU     256     ; size of internal buffers for text i/o
P_SAFESIZE      EQU     128     ; safety margin for text output buffer

*------ pd_Flags ------
   BITDEF   P,IOR0,0            ; IOR0 is in use
   BITDEF   P,IOR1,1            ; IOR1 is in use
   BITDEF   P,EXPUNGED,7        ; device to be expunged when all closed

 STRUCTURE  PrinterData,dd_SIZEOF
    STRUCT  pd_Unit,MP_SIZE     ; the one and only unit
    BPTR pd_PrinterSegment      ; the printer specific segment
    UWORD   pd_PrinterType      ; the segment printer type
    APTR pd_SegmentData         ; the segment data structure
    APTR pd_PrintBuf            ; the raster print buffer
    APTR pd_PWrite              ; the parallel write function
    APTR pd_PBothReady          ; the parallel write function's done

    IFGT IOEXTPar_SIZE-IOEXTSER_SIZE
    STRUCT  pd_IOR0,IOEXTPar_SIZE   ; port I/O request 0
    STRUCT  pd_IOR1,IOEXTPar_SIZE   ;   and 1 for double buffering
    ENDC

    IFLE IOEXTPar_SIZE-IOEXTSER_SIZE
    STRUCT  pd_IOR0,IOEXTSER_SIZE   ; port I/O request 0
    STRUCT  pd_IOR1,IOEXTSER_SIZE   ;   and 1 for double buffering
    ENDC

    STRUCT  pd_TIOR,IOTV_SIZE       ; timer I/O request
    STRUCT  pd_IORPort,MP_SIZE      ;   and message reply port
    STRUCT  pd_TC,TC_SIZE           ; write task
    STRUCT  pd_OldStk,P_OLDSTKSIZE  ;   and stack space (OBSOLETE)
    UBYTE   pd_Flags                ; device flags
    UBYTE   pd_pad                  ; padding
    STRUCT  pd_Preferences,pf_SIZEOF ; the latest preferences
    UBYTE   pd_PWaitEnabled         ; wait function switch
;   /* new fields for V2.0 */
    UBYTE   pd_Pad1                 ; padding
    STRUCT  pd_Stk,P_STKSIZE        ; stack space
    LABEL   pd_SIZEOF               ; warning! this may be odd

    BITDEF  PPC,GFX,0           ;graphics (bit position)
    BITDEF  PPC,COLOR,1         ;color (bit position)

PPC_BWALPHA     EQU     $00     ;black&white alphanumerics
PPC_BWGFX       EQU     $01     ;black&white graphics
PPC_COLORALPHA  EQU     $02     ;color alphanumerics
PPC_COLORGFX    EQU     $03     ;color graphics

PCC_BW          EQU     1       ;black&white only
PCC_YMC         EQU     2       ;yellow/magenta/cyan only
PCC_YMC_BW      EQU     3       ;yellow/magenta/cyan or black&white
PCC_YMCB        EQU     4       ;yellow/magenta/cyan/black

PCC_4COLOR      EQU     $4      ;a flag for YMCB and BGRW
```

devices/prtbase.i

```
PCC_ADDITIVE    EQU     $8      ;not ymcb but blue/green/red/white
PCC_WB          EQU     $9      ;black&white only, 0 == BLACK
PCC_BGR         EQU     $a      ;blue/green/red
PCC_BGR_WB      EQU     $b      ;blue/green/red or black&white
PCC_BGRW        EQU     $c      ;blue/green/red/white
;       The picture must be scanned once for each color component, as the
;       printer can only define one color at a time.  ie. If 'PCC_YMC' then
;       first pass sends all 'Y' info to printer, second pass sends all 'M'
;       info, and third pass sends all C info to printer.  The CalComp
;       PlotMaster is an example of this type of printer.
PCC_MULTI_PASS  EQU     $10     ;see explanation above

 STRUCTURE  PrinterExtendedData,0
    APTR    ped_PrinterName   ; printer name, null terminated
    APTR    ped_Init          ; called after LoadSeg
    APTR    ped_Expunge       ; called before UnLoadSeg
    APTR    ped_Open          ; called at OpenDevice
    APTR    ped_Close         ; called at CloseDevice
    UBYTE   ped_PrinterClass  ; printer class
    UBYTE   ped_ColorClass    ; color class
    UBYTE   ped_MaxColumns    ; number of print columns available
    UBYTE   ped_NumCharSets   ; number of character sets
    UWORD   ped_NumRows       ; number of 'pins' in print head
    ULONG   ped_MaxXDots      ; number of dots maximum in a raster dump
    ULONG   ped_MaxYDots      ; number of dots maximum in a raster dump
    UWORD   ped_XDotsInch     ; horizontal dot density
    UWORD   ped_YDotsInch     ; vertical dot density
    APTR    ped_Commands      ; printer text command table
    APTR    ped_DoSpecial     ; special command handler
    APTR    ped_Render        ; raster render function
    LONG    ped_TimeoutSecs   ; good write timeout
;------ the following only exists if the segment version is 33 or greater
        APTR     ped_8BitChars  ;conversion strings for the extended font
        LONG     ped_PrintMode  ;set if text printed, otherwise 0
;------ the following only exists if the segment version is 34 or greater
        APTR    ped_ConvFunv    ; ptr to conversion function for all chars
        LABEL   ped_SIZEOF

 STRUCTURE  PrinterSegment,0
    ULONG   ps_NextSegment    ; (actually a BPTR)
    ULONG   ps_runAlert       ; MOVEQ #0,D0 : RTS
    UWORD   ps_Version        ; segment version
    UWORD   ps_Revision       ; segment revision
    LABEL   ps_PED            ; printer extended data

  ENDC
```

devices/prtgfx.i

```
        IFND    DEVICES_PRTGFX_I
DEVICES_PRTGFX_I        SET     1
**
**      $Filename: devices/prtgfx.i $
**      $Release: 2.04 $
**      $Revision: 1.12 $
**      $Date: 90/07/26 $
**
**      printer.device structure definitions
**
**      (C) Copyright 1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

    IFND  EXEC_TYPES_I
    INCLUDE  "exec/types.i"
    ENDC

PCMYELLOW       EQU     0               ; byte index for yellow
PCMMAGENTA      EQU     1               ; byte index for magenta
PCMCYAN         EQU     2               ; byte index for cyan
PCMBLACK        EQU     3               ; byte index for black
PCMBLUE         EQU     PCMYELLOW       ; byte index for blue
PCMGREEN        EQU     PCMMAGENTA      ; byte index for green
PCMRED          EQU     PCMCYAN         ; byte index for red
PCMWHITE        EQU     PCMBLACK        ; byte index for white

        STRUCTURE       colorEntry,0
                LABEL   colorLong       ; quick access to all of YMCB
                LABEL   colorSByte      ; 1 entry for each of YMCB
                STRUCT  colorByte,4     ; ditto (except signed)
                LABEL   ce_SIZEOF

        STRUCTURE       PrtInfo,0
                APTR    pi_render       ; PRIVATE - DO NOT USE!
                APTR    pi_rp           ; PRIVATE - DO NOT USE!
                APTR    pi_temprp       ; PRIVATE - DO NOT USE!
                APTR    pi_RowBuf       ; PRIVATE - DO NOT USE!
                APTR    pi_HamBuf       ; PRIVATE - DO NOT USE!
                APTR    pi_ColorMap     ; PRIVATE - DO NOT USE!
                APTR    pi_ColorInt     ; color intensities for entire row
                APTR    pi_HamInt       ; PRIVATE - DO NOT USE!
                APTR    pi_Dest1Int     ; PRIVATE - DO NOT USE!
                APTR    pi_Dest2Int     ; PRIVATE - DO NOT USE!
                APTR    pi_ScaleX       ; array of scale values for X
                APTR    pi_ScaleXAlt    ; PRIVATE - DO NOT USE!
                APTR    pi_dmatrix      ; pointer to dither matrix
                APTR    pi_TopBuf       ; PRIVATE - DO NOT USE!
                APTR    pi_BotBuf       ; PRIVATE - DO NOT USE!

                UWORD   pi_RowBufSize   ; PRIVATE - DO NOT USE!
                UWORD   pi_HamBufSize   ; PRIVATE - DO NOT USE!
                UWORD   pi_ColorMapSize ; PRIVATE - DO NOT USE!
                UWORD   pi_ColorIntSize ; PRIVATE - DO NOT USE!
                UWORD   pi_HamIntSize   ; PRIVATE - DO NOT USE!
                UWORD   pi_Dest1IntSize ; PRIVATE - DO NOT USE!
                UWORD   pi_Dest2IntSize ; PRIVATE - DO NOT USE!
                UWORD   pi_ScaleXSize   ; PRIVATE - DO NOT USE!
                UWORD   pi_ScaleXAltSize ; PRIVATE - DO NOT USE!

                UWORD   pi_PrefsFlags   ; PRIVATE - DO NOT USE!
                ULONG   pi_special      ; PRIVATE - DO NOT USE!
                UWORD   pi_xstart       ; PRIVATE - DO NOT USE!
                UWORD   pi_ystart       ; PRIVATE - DO NOT USE!
                UWORD   pi_width        ; source width (in pixels)
                UWORD   pi_height       ; PRIVATE - DO NOT USE!
```

devices/prtgfx.i

```
                ULONG   pi_pc           ; PRIVATE - DO NOT USE!
                ULONG   pi_pr           ; PRIVATE - DO NOT USE!
                UWORD   pi_ymult        ; PRIVATE - DO NOT USE!
                UWORD   pi_ymod         ; PRIVATE - DO NOT USE!
                UWORD   pi_ety          ; PRIVATE - DO NOT USE!
                UWORD   pi_xpos         ; offset to start printing from
                UWORD   pi_threshold    ; copy of threshold value (from prefs)
                UWORD   pi_tempwidth    ; PRIVATE - DO NOT USE!
                UWORD   pi_flags        ; PRIVATE - DO NOT USE!
                LABEL   prtinfo_SIZEOF

        ENDC    ; DEVICES_PRTGFX_I
```

devices/scsidisk.i

```
        IFND    DEVICES_SCSIDISK_I
DEVICES_SCSIDISK_I      EQU     1
**
**      $Filename: devices/scsidisk.i $
**      $Release: 2.04 $
**      $Revision: 36.2 $
**      $Date: 90/11/07 $
**
**      SCSI exec-level device command
**
**      (C) Copyright 1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**


    IFND EXEC_TYPES_I
    INCLUDE "exec/types.i"
    ENDC ; EXEC_TYPES_I


;---------------------------------------------------------------------
;
;   SCSI Command
;       Several Amiga SCSI controller manufacturers are converging on
;       standard ways to talk to their controllers.  This include
;       file describes an exec-device command (e.g. for hddisk.device)
;       that can be used to issue SCSI commands
;
;   UNIT NUMBERS
;       Unit numbers to the OpenDevice call have encoded in them which
;       SCSI device is being referred to.  The three decimal digits of
;       the unit number refer to the SCSI Target ID (bus address) in
;       the 1's digit, the SCSI logical unit (LUN) in the 10's digit,
;       and the controller board in the 100's digit.
;
;       Examples:
;                 0     drive at address 0
;                12     LUN 1 on multiple drive controller at address 2
;               104     second controller board, address 4
;                88     not valid: both logical units and addresses
;                       range from 0..7.
;
;   CAVEATS
;       Original 2090 code did not support this command.
;
;       Commodore 2090/2090A unit numbers are different.  The SCSI
;       logical unit is the 100's digit, and the SCSI Target ID
;       is a permuted 1's digit: Target ID 0..6 maps to unit 3..9
;       (7 is reserved for the controller).
;
;           Examples:
;                 3     drive at address 0
;               109     drive at address 6, logical unit 1
;                 1     not valid: this is not a SCSI unit.  Perhaps
;                       it's an ST506 unit.
;
;       Some controller boards generate a unique name (e.g. 2090A's
;       iddisk.device) for the second controller board, instead of
;       implementing the 100's digit.
;
;       There are optional restrictions on the alignment, bus
;       accessability, and size of the data for the data phase.
;       Be conservative to work with all manufacturer's controllers.
;
;---------------------------------------------------------------------
```

devices/scsidisk.i Page 2

```
HD_SCSICMD      EQU     28          ; issue a SCSI command to the unit
                                    ; io_Data points to a SCSICmd
                                    ; io_Length is sizeof(struct SCSICmd)
                                    ; io_Actual and io_Offset are not used

 STRUCTURE      SCSICmd,0
    APTR    scsi_Data               ; word aligned data for SCSI Data Phase
                                    ; (optional) data need not be byte aligned
                                    ; (optional) data need not be bus accessable
    ULONG   scsi_Length             ; even length of Data area
                                    ; (optional) data can have odd length
                                    ; (optional) data length can be > 2**24
    ULONG   scsi_Actual             ; actual Data used
    APTR    scsi_Command            ; SCSI Command (same options as scsi_Data)
    UWORD   scsi_CmdLength          ; length of Command
    UWORD   scsi_CmdActual          ; actual Command used
    UBYTE   scsi_Flags              ; includes intended data direction
    UBYTE   scsi_Status             ; SCSI status of command
    APTR    scsi_SenseData          ; sense data: filled if SCSIF_[OLD]AUTOSENSE
                                    ; is set and scsi_Status has CHECK CONDITION
                                    ; (bit 1) set
    UWORD   scsi_SenseLength        ; size of scsi_SenseData, also bytes to
                                    ; request w/ SCSIF_AUTOSENSE, must be 4..255
    UWORD   scsi_SenseActual        ; amount actually fetched (0 means no sense)
    LABEL   scsi_SIZEOF


;------ scsi_Flags ------
SCSIF_WRITE             EQU     0       ; intended data direction is out
SCSIF_READ              EQU     1       ; intended data direction is in
SCSIB_READ_WRITE        EQU     0       ; (the bit to test)

SCSIF_NOSENSE           EQU     0       ; no automatic request sense
SCSIF_AUTOSENSE         EQU     2       ; do standard extended request sense
                                        ; on check condition
SCSIF_OLDAUTOSENSE      EQU     6       ; do 4 byte non-extended request
                                        ; sense on check condition
SCSIB_AUTOSENSE         EQU     1       ; (the bit to test)
SCSIB_OLDAUTOSENSE      EQU     2       ; (the bit to test)

;------ SCSI io_Error values ------
HFERR_SelfUnit          EQU     40      ; cannot issue SCSI command to self
HFERR_DMA               EQU     41      ; DMA error
HFERR_Phase             EQU     42      ; illegal or unexpected SCSI phase
HFERR_Parity            EQU     43      ; SCSI parity error
HFERR_SelTimeout        EQU     44      ; Select timed out
HFERR_BadStatus         EQU     45      ; status and/or sense error

;------ OpenDevice io_Error values ------
HFERR_NoBoard           EQU     50      ; Open failed for non-existant board

        ENDC    ; DEVICES_SCSIDISK_I
```

devices/serial.i Page 1

```
        IFND DEVICES_SERIAL_I
DEVICES_SERIAL_I SET 1
**
**      $Filename: devices/serial.i $
**      $Release: 2.04 $
**      $Revision: 33.6 $
**      $Date: 90/11/06 $
**
**      external declarations for the serial device
**
**      (C) Copyright 1985,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**
    IFND    EXEC_IO_I
    include "exec/io.i"
    ENDC    !EXEC_IO_I

*----------------------------------------------------------------------
*
* Useful constants
*
*----------------------------------------------------------------------
*
SER_DEFAULT_CTLCHAR EQU $11130000  ; default chars for xON,xOFF
; You may change these via SETPARAMS.  At this time, parity is not
; calculated for xON/xOFF characters.  You must supply them with the
; desired parity.

*
*----------------------------------------------------------------------
*
* Driver Specific Commands

SDCMD_QUERY     EQU     CMD_NONSTD      ;$09
SDCMD_BREAK     EQU     CMD_NONSTD+1    ;$0A
SDCMD_SETPARAMS EQU     CMD_NONSTD+2    ;$0B

SER_DEVFINISH   EQU     CMD_NONSTD+2 ; number of device comands

*----------------------------------------------------------------------

SERIALNAME:     MACRO
                dc.b    'serial.device',0
                dc.w    0
                ENDM

        BITDEF  SER,XDISABLED,7   ; SERFLAGS xOn-xOff feature disabled bit
        BITDEF  SER,EOFMODE,6     ;    "     EOF mode enabled bit
        BITDEF  SER,SHARED,5      ;    "     non-exclusive access
        BITDEF  SER,RAD_BOOGIE,4  ;    "     high-speed mode active
        BITDEF  SER,QUEUEDBRK,3   ;    "     queue this Break ioRqst
        BITDEF  SER,7WIRE,2       ;    "     RS232 7-wire protocol
        BITDEF  SER,PARTY_ODD,1   ;    "     use-odd-parity bit
        BITDEF  SER,PARTY_ON,0    ;    "     parity-enabled bit
;
;WARNING: The next series of BITDEFs refer to the HIGH order BYTE of
;IO_STATUS.  Example usage: "BTST.B #IOST_XOFFWRITE,IO_STATUS+1(AX)"
;
        BITDEF  IOST,XOFFREAD,4   ; IOST_HOB receive currently xOFF'ed
        BITDEF  IOST,XOFFWRITE,3  ;    "     transmit currently xOFF'ed
        BITDEF  IOST,READBREAK,2  ;    "     break was latest input
        BITDEF  IOST,WROTEBREAK,1 ;    "     break was latest output
        BITDEF  IOST,OVERRUN,0    ;    "     status word RBF overrun
;
;       BITDEF's in a longword field)
;       Example usage: BSET.B #SEXTB_MSPON,IO_EXTFLAGS+3(AX)
```

```
                                ;IO_EXTFLAGS (extended flag longword)
        BITDEF  SEXT,MSPON,1    ;    "    use mark-space parity,not odd-even
        BITDEF  SEXT,MARK,0     ;    "    if mark-space, use mark
*
*****************************************************************************
 STRUCTURE TERMARRAY,0
        ULONG    TERMARRAY_0
        ULONG    TERMARRAY_1
        LABEL    TERMARRAY_SIZE

****************************************************************
* CAUTION !!  IF YOU ACCESS the serial.device, you MUST (!!!!) use an
* IOEXTSER-sized structure or you may overlay innocent memory, okay ?!
****************************************************************

 STRUCTURE IOEXTSER,IOSTD_SIZE

*    STRUCT   MsgNode
*   0   APTR     Succ
*   4   APTR     Pred
*   8   UBYTE    Type
*   9   UBYTE    Pri
*   A   APTR     Name
*   E   APTR     ReplyPort
*  12   UWORD    MNLength
*    STRUCT   IOExt
*  14   APTR     IO_DEVICE
*  18   APTR     IO_UNIT
*  1C   UWORD    IO_COMMAND
*  1E   UBYTE    IO_FLAGS
*  1F   UBYTE    IO_ERROR
*    STRUCT   IOStdExt
*  20   ULONG    IO_ACTUAL
*  24   ULONG    IO_LENGTH
*  28   APTR     IO_DATA
*  2C   ULONG    IO_OFFSET
*
*  30
        ULONG   IO_CTLCHAR      ; control char's (order = xON,xOFF,rsvd,rsvd)
        ULONG   IO_RBUFLEN      ; length in bytes of serial port's read buffer
        ULONG   IO_EXTFLAGS     ; additional serial flags (see bitdefs above)
        ULONG   IO_BAUD         ; baud rate requested (true baud)
        ULONG   IO_BRKTIME      ; duration of break signal in MICROseconds
        STRUCT  IO_TERMARRAY,TERMARRAY_SIZE ; termination character array
        UBYTE   IO_READLEN      ; bits per read char (bit count)
        UBYTE   IO_WRITELEN     ; bits per write char (bit count)
        UBYTE   IO_STOPBITS     ; stopbits for read (count)
        UBYTE   IO_SERFLAGS     ; see SERFLAGS bit definitions above
        UWORD   IO_STATUS       ; status of serial port, as follows:
*
*                 BIT  ACTIVE  FUNCTION
*                  0    ---    reserved
*                  1    ---    reserved
*                  2    high   Connected to parallel "select" on the A1000.
*                              Connected to both the parallel "select" and
*                              serial "ring indicator" pins on the A500
*                              & A2000.  Take care when making cables.
*                  3    low    Data Set Ready
*                  4    low    Clear To Send
*                  5    low    Carrier Detect
*                  6    low    Ready To Send
*                  7    low    Data Terminal Ready
*                  8    high   read overrun
*                  9    high   break sent
*                 10    high   break received
*                 11    high   transmit x-OFF'ed
```

```
*                  12    high   receive x-OFF'ed
*               13-15           reserved
*
        LABEL   IOEXTSER_SIZE

*****************************************************************************

*--------------------------------------------------------------------
*
* Driver error definitions
*
*--------------------------------------------------------------------

SerErr_DevBusy          EQU     1
SerErr_BaudMismatch     EQU     2       ;baud rate not supported by hardware
SerErr_BufErr           EQU     4       ;Failed to allocate new read buffer
SerErr_InvParam         EQU     5
SerErr_LineErr          EQU     6
SerErr_ParityErr        EQU     9
SerErr_TimerErr         EQU     11      ;(See the serial/OpenDevice autodoc)
SerErr_BufOverflow      EQU     12
SerErr_NoDSR            EQU     13
SerErr_DetectedBreak    EQU     15


 IFD    DEVICES_SERIAL_I_OBSOLETE
SER_DBAUD               EQU     9600    ;unused
SerErr_InvBaud          EQU     3       ;unused
SerErr_NotOpen          EQU     7       ;unused
SerErr_PortReset        EQU     8       ;unused
SerErr_InitErr          EQU     10      ;unused
SerErr_NoCTS            EQU     14      ;unused
        BITDEF  IOSER,QUEUED,6    ; IO_FLAGS rqst-queued bit
        BITDEF  IOSER,ABORT,5     ;    "    rqst-aborted bit
        BITDEF  IOSER,ACTIVE,4    ;    "    rqst-qued-or-current bit
 ENDC


    ENDC    !DEVICES_SERIAL_I
```

devices/timer.i

```
        IFND    DEVICES_TIMER_I
DEVICES_TIMER_I SET     1
**
**      $Filename: devices/timer.i $
**      $Release: 2.04 $
**      $Revision: 36.10 $
**      $Date: 91/03/05 $
**
**      Timer device name and useful definitions.
**
**      (C) Copyright 1985,1985,1987,1988,1989,1990 Commodore-Amiga Inc.
**              All Rights Reserved
**


        IFND    EXEC_TYPES_I
        INCLUDE "exec/types.i"
        ENDC

        IFND    EXEC_IO_I
        INCLUDE "exec/io.i"
        ENDC

* unit defintions
UNIT_MICROHZ    EQU     0
UNIT_VBLANK     EQU     1
UNIT_ECLOCK     EQU     2
UNIT_WAITUNTIL  EQU     3
UNIT_WAITECLOCK EQU     4

TIMERNAME       MACRO
                DC.B    'timer.device',0
                DS.W    0
                ENDM

 STRUCTURE TIMEVAL,0
        ULONG   TV_SECS
        ULONG   TV_MICRO
        LABEL   TV_SIZE

 STRUCTURE ECLOCKVAL,0
        ULONG   EV_HI
        ULONG   EV_LO
        LABEL   EV_SIZE

 STRUCTURE TIMEREQUEST,IO_SIZE
        STRUCT  IOTV_TIME,TV_SIZE
        LABEL   IOTV_SIZE

* IO_COMMAND to use for adding a timer
        DEVINIT
        DEVCMD  TR_ADDREQUEST
        DEVCMD  TR_GETSYSTIME
        DEVCMD  TR_SETSYSTIME

        ENDC    ; DEVICES_TIMER_I
```

devices/trackdisk.i

```
        IFND    DEVICES_TRACKDISK_I
DEVICES_TRACKDISK_I     SET     1

**
**      $Filename: devices/trackdisk.i $
**      $Release: 2.04 $
**      $Revision: 33.12 $
**      $Date: 90/11/28 $
**
**      trackdisk device structure and value definitions
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

        IFND    EXEC_IO_I
        INCLUDE "exec/io.i"
        ENDC    !EXEC_IO_I

        IFND    EXEC_DEVICES_I
        INCLUDE "exec/devices.i"
        ENDC    !EXEC_DEVICES_I

*--------------------------------------------------------------------
*
* Physical drive constants
*
*--------------------------------------------------------------------


* OBSOLETE -- only valid for 3 1/4" drives.  Use the TD_GETNUMTRACKS command!
*
*NUMCYLS                        EQU     80              ; normal # of cylinders
*MAXCYLS                        EQU     NUMCYLS+20      ; max # of cyls to look for
*                                                       ;       during a calibrate
*NUMHEADS               EQU     2
*NUMTRACKS              EQU     NUMCYLS*NUMHEADS

NUMSECS                 EQU     11
NUMUNITS                EQU     4

*-------------------------------------------------------------- ---
*
* Useful constants
*
*--------------------------------------------------------------------


*-- sizes before mfm encoding
TD_SECTOR               EQU     512
TD_SECSHIFT             EQU     9                       ; log TD_SECTOR
*                                                       ;     2


*--------------------------------------------------------------------
*
* Driver Specific Commands
*
*--------------------------------------------------------------------

*-- TD_NAME is a generic macro to get the name of the driver.  This
*-- way if the name is ever changed you will pick up the change
*-- automatically.
*--
*-- Normal usage would be:
*--
```

```
*-- internalName:       TD_NAME
*--

TD_NAME:        MACRO
                DC.B    'trackdisk.device',0
                DS.W    0
                ENDM

        BITDEF  TD,EXTCOM,15

        DEVINIT
        DEVCMD  TD_MOTOR                ; control the disk's motor
        DEVCMD  TD_SEEK                 ; explicit seek (for testing)
        DEVCMD  TD_FORMAT               ; format disk
        DEVCMD  TD_REMOVE               ; notify when disk changes
        DEVCMD  TD_CHANGENUM            ; number of disk changes
        DEVCMD  TD_CHANGESTATE          ; is there a disk in the drive?
        DEVCMD  TD_PROTSTATUS           ; is the disk write protected?
        DEVCMD  TD_RAWREAD              ; read raw bits from the disk
        DEVCMD  TD_RAWWRITE             ; write raw bits to the disk
        DEVCMD  TD_GETDRIVETYPE         ; get the type of the disk drive
        DEVCMD  TD_GETNUMTRACKS         ; get the # of tracks on this disk
        DEVCMD  TD_ADDCHANGEINT         ; TD_REMOVE done right
        DEVCMD  TD_REMCHANGEINT         ; removes softint set by ADDCHANGEINT
        DEVCMD  TD_GETGEOMETRY          ; gets the disk geometry table
        DEVCMD  TD_EJECT                ; for those drives that support it
        DEVCMD  TD_LASTCOMM             ; dummy placeholder for end of list


*
*
* The disk driver has an "extended command" facility.  These commands
* take a superset of the normal IO Request block.
*
ETD_WRITE       EQU     (CMD_WRITE!TDF_EXTCOM)
ETD_READ        EQU     (CMD_READ!TDF_EXTCOM)
ETD_MOTOR       EQU     (TD_MOTOR!TDF_EXTCOM)
ETD_SEEK        EQU     (TD_SEEK!TDF_EXTCOM)
ETD_FORMAT      EQU     (TD_FORMAT!TDF_EXTCOM)
ETD_UPDATE      EQU     (CMD_UPDATE!TDF_EXTCOM)
ETD_CLEAR       EQU     (CMD_CLEAR!TDF_EXTCOM)
ETD_RAWREAD     EQU     (TD_RAWREAD!TDF_EXTCOM)
ETD_RAWWRITE    EQU     (TD_RAWWRITE!TDF_EXTCOM)


*
* extended IO has a larger than normal io request block.
*

 STRUCTURE IOEXTTD,IOSTD_SIZE
        ULONG   IOTD_COUNT      ; removal/insertion count
        ULONG   IOTD_SECLABEL   ; sector label data region
        LABEL   IOTD_SIZE


*
*  This is the structure returned by TD_DRIVEGEOMETRY
*  Note that the layout can be defined three ways:
*
*  1. TotalSectors
*  2. Cylinders and CylSectors
*  3. Cylinders, Heads, and TrackSectors.
*
*  #1 is most accurate, #2 is less so, and #3 is least accurate.  All
*  are usable, though #2 and #3 may waste some portion of the available
*  space on some drives.
```

```
*

 STRUCTURE DriveGeometry,0
        ULONG   dg_SectorSize           ; in bytes
        ULONG   dg_TotalSectors         ; total # of sectors on drive
        ULONG   dg_Cylinders            ; number of cylinders
        ULONG   dg_CylSectors           ; number of sectors/cylinder
        ULONG   dg_Heads                ; number of surfaces
        ULONG   dg_TrackSectors         ; number of sectors/track
        ULONG   dg_BufMemType           ; preferred buffer memory type
                                        ; (usually MEMF_PUBLIC)
        UBYTE   dg_DeviceType           ; codes as defined in the SCSI-2 spec
        UBYTE   dg_Flags                ; flags, including removable
        UWORD   dg_Reserved
 LABEL dg_SIZEOF

* device types
DG_DIRECT_ACCESS        EQU     0
DG_SEQUENTIAL_ACCESS    EQU     1
DG_PRINTER              EQU     2
DG_PROCESSOR            EQU     3
DG_WORM                 EQU     4
DG_CDROM                EQU     5
DG_SCANNER              EQU     6
DG_OPTICAL_DISK         EQU     7
DG_MEDIUM_CHANGER       EQU     8
DG_COMMUNICATION        EQU     9
DG_UNKNOWN              EQU     31

* flags
 BITDEF DG,REMOVABLE,0


*
* raw read and write can be synced with the index pulse.  This flag
* in io request's IO_FLAGS field tells the driver that you want this.
*
        BITDEF  IOTD,INDEXSYNC,4
*
* raw read and write can be synced with a $4489 sync pattern.  This flag
* in io request's IO_FLAGS field tells the driver that you want this.
*
        BITDEF  IOTD,WORDSYNC,5

* labels are TD_LABELSIZE bytes per sector

TD_LABELSIZE    EQU     16

*
* This is a bit in the FLAGS field of OpenDevice.  If it is set, then
* the driver will allow you to open all the disks that the trackdisk
* driver understands.  Otherwise only 3.5" disks will succeed.
*
*
        BITDEF  TD,ALLOW_NON_3_5,0

*
*  If you set the TDB_ALLOW_NON_3_5 bit in OpenDevice, then you don't
*  know what type of disk you really got.  These defines are for the
*  TD_GETDRIVETYPE command.  In addition, you can find out how many
*  tracks are supported via the TD_GETNUMTRACKS command.
*
DRIVE3_5        EQU     1
DRIVE5_25       EQU     2
DRIVE3_5_150RPM EQU     3

```

devices/trackdisk.i

```
*-------------------------------------------------------------------
*
* Driver error defines
*
*-------------------------------------------------------------------

TDERR_NotSpecified      EQU     20      ; general catchall
TDERR_NoSecHdr          EQU     21      ; couldn't even find a sector
TDERR_BadSecPreamble    EQU     22      ; sector looked wrong
TDERR_BadSecID          EQU     23      ; ditto
TDERR_BadHdrSum         EQU     24      ; header had incorrect checksum
TDERR_BadSecSum         EQU     25      ; data had incorrect checksum
TDERR_TooFewSecs        EQU     26      ; couldn't find enough sectors
TDERR_BadSecHdr         EQU     27      ; another "sector looked wrong"
TDERR_WriteProt         EQU     28      ; can't write to a protected disk
TDERR_DiskChanged       EQU     29      ; no disk in the drive
TDERR_SeekError         EQU     30      ; couldn't find track 0
TDERR_NoMem             EQU     31      ; ran out of memory
TDERR_BadUnitNum        EQU     32      ; asked for a unit > NUMUNITS
TDERR_BadDriveType      EQU     33      ; not a drive that trackdisk groks
TDERR_DriveInUse        EQU     34      ; someone else allocated the drive
TDERR_PostReset         EQU     35      ; user hit reset; awaiting doom

*-------------------------------------------------------------------
*
* Public portion of unit structure
*
*-------------------------------------------------------------------

 STRUCTURE TDU_PUBLICUNIT,UNIT_SIZE
        UWORD   TDU_COMP01TRACK         ; track for first precomp
        UWORD   TDU_COMP10TRACK         ; track for second precomp
        UWORD   TDU_COMP11TRACK         ; track for third precomp
        ULONG   TDU_STEPDELAY           ; time to wait after stepping
        ULONG   TDU_SETTLEDELAY         ; time to wait after seeking
        UBYTE   TDU_RETRYCNT            ; # of times to retry
        UBYTE   TDU_PUBFLAGS            ; public flags, see below
        UWORD   TDU_CURRTRK             ; track heads are over
                                        ;  (ONLY ACCESS WHILE UNIT IS STOPPED!)
        ULONG   TDU_CALIBRATEDELAY      ; time to wait after stepping
                                        ; for recalibrate
        ULONG   TDU_COUNTER             ; counter for disk changes
                                        ;  (ONLY ACCESS WHILE UNIT IS STOPPED!)
        LABEL   TDU_PUBLICUNITSIZE


*
* Flags for TDU_PUBFLAGS:
*
        BITDEF  TDP,NOCLICK,0           ; set to enable noclickstart

        ENDC    ; DEVICE_TRACKDISK_I
```

dos/datetime.i

Page 1

```
        IFND    DOS_DATETIME_I
DOS_DATETIME_I SET 1

**
**      $Filename: dos/datetime.i $
**      $Release: 2.04 $
**      $Revision: 36.7 $
**      $Date: 90/07/12 $
**
**      Date and time assembler header for AmigaDOS
**
**      (C) Copyright 1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**

        IFND DOS_DOS_I
        INCLUDE "dos/dos.i"
        ENDC

*
*       Data structures and equates used by the V1.4 DOS functions
* StrtoDate() and DatetoStr()
*
*

*--------- String/Date structures etc
    STRUCTURE   DateTime,0
        STRUCT  dat_Stamp,ds_SIZEOF     ;DOS DateStamp
        UBYTE   dat_Format              ;controls appearance of dat_StrDate
        UBYTE   dat_Flags               ;see BITDEF's below
        CPTR    dat_StrDay              ;day of the week string
        CPTR    dat_StrDate             ;date string
        CPTR    dat_StrTime             ;time string
        LABEL   dat_SIZEOF
*
* You need this much room for each of the DateTime strings:
LEN_DATSTRING   EQU     16

*       flags for dat_Flags
*
        BITDEF  DT,SUBST,0              ;substitute Today, Tomorrow, etc.
        BITDEF  DT,FUTURE,1             ;day of the week is in future
*
*       date format values
*
FORMAT_DOS      equ     0               ; dd-mmm-yy
FORMAT_INT      equ     1               ; yy-mm-dd
FORMAT_USA      equ     2               ; mm-dd-yy
FORMAT_CDN      equ     3               ; dd-mm-yy
FORMAT_MAX      equ     FORMAT_CDN

        ENDC    ; DOS_DATETIME_I
```

dos/dos.i

Page 1

```
        IFND    DOS_DOS_I
DOS_DOS_I SET   1
**      $Filename: dos/dos.i $
**      $Release: 2.04 $
**      $Revision: 36.20 $
**      $Date: 91/02/13 $
**
**      Standard asm header for AmigaDOS
**
**      (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**

        IFND    EXEC_TYPES_I
        INCLUDE "exec/types.i"
        ENDC

DOSNAME     MACRO
      DC.B  'dos.library',0
        ENDM

* Predefined Amiga DOS global constants

DOSTRUE         EQU     -1
DOSFALSE        EQU      0

* Mode parameter to Open()
MODE_OLDFILE            EQU   1005    * Open existing file read/write
*                                     * positioned at beginning of file.
MODE_NEWFILE            EQU   1006    * Open freshly created file (delete
*                                     * old file) read/write
MODE_READWRITE          EQU   1004    * Open old file w/shared lock,
*                                     * creates file if doesn't exist.

* Relative position to Seek()
OFFSET_BEGINNING        EQU   -1      * relative to Beginning Of File
OFFSET_CURRENT          EQU    0      * relative to Current file position
OFFSET_END              EQU    1      * relative to End Of File

OFFSET_BEGINING        EQU   OFFSET_BEGINNING     * Ancient compatibility

BITSPERBYTE             EQU   8
BYTESPERLONG            EQU   4
BITSPERLONG             EQU   32
MAXINT                  EQU   $7FFFFFFF
MININT                  EQU   $80000000

* Passed as type to Lock()
SHARED_LOCK             EQU   -2    ; File is readable by others
ACCESS_READ             EQU   -2    ; Synonym
EXCLUSIVE_LOCK          EQU   -1    ; No other access allowed
ACCESS_WRITE            EQU   -1    ; Synonym


 STRUCTURE DateStamp,0
   LONG  ds_Days                    ; Number of days since Jan. 1, 1978
   LONG  ds_Minute                  ; Number of minutes past midnight
   LONG  ds_Tick                    ; Number of ticks past minute
   LABEL ds_SIZEOF                  ; DateStamp

TICKS_PER_SECOND EQU 50             ; Number of ticks in one second

* Returned by Examine() and ExInfo()
 STRUCTURE FileInfoBlock,0
   LONG   fib_DiskKey
   LONG   fib_DirEntryType          ; Type of Directory. If < 0, then a plain file.
```

dos/dos.i Page 2

```
                                  ; If > 0 a directory
    STRUCT fib_FileName,108       ; Null terminated. Max 30 chars used for now
    LONG   fib_Protection         ; bit mask of protection, rwxd are 3-0.
    LONG   fib_EntryType
    LONG   fib_Size               ; Number of bytes in file
    LONG   fib_NumBlocks          ; Number of blocks in file
    STRUCT fib_DateStamp,ds_SIZEOF ; Date file last changed.
    STRUCT fib_Comment,80         ; Null terminated. Comment associated with file
    STRUCT fib_Reserved,36
    LABEL  fib_SIZEOF             ; FileInfoBlock

* FIB stands for FileInfoBlock
* FIBB are bit definitions, FIBF are field definitions
    BITDEF   FIB,SCRIPT,6         ; program is an execute script
    BITDEF   FIB,PURE,5           ; program is reentrant and reexecutable
    BITDEF   FIB,ARCHIVE,4        ; cleared whenever file is changed
    BITDEF   FIB,READ,3           ; ignored by the system
    BITDEF   FIB,WRITE,2          ; ignored by the system
    BITDEF   FIB,EXECUTE,1        ; ignored by the system
    BITDEF   FIB,DELETE,0         ; prevent file from being deleted

* Standard maximum length for an error string from fault.  However, most
* error strings should be kept under 60 characters if possible.  Don't
* forget space for the header you pass in.
FAULT_MAX       EQU     82

* All BCPL data must be long word aligned.  BCPL pointers are the long word
* address (i.e byte address divided by 4 (>>2))

* Macro to indicate BCPL pointers
BPTR     MACRO                        * Long word pointer
         LONG     \1
         ENDM
BSTR     MACRO                        * Long word pointer to BCPL string.
         LONG     \1
         ENDM

* #define BADDR( bptr ) (bptr << 2) * Convert BPTR to byte addressed pointer

* BCPL strings have a length in the first byte and then the characters.
* For example:  s[0]=3 s[1]=S s[2]=Y s[3]=S

* returned by Info()
  STRUCTURE InfoData,0
    LONG id_NumSoftErrors         * number of soft errors on disk
    LONG id_UnitNumber            * Which unit disk is (was) mounted on
    LONG id_DiskState             * See defines below
    LONG id_NumBlocks             * Number of blocks on disk
    LONG id_NumBlocksUsed         * Number of block in use
    LONG id_BytesPerBlock
    LONG id_DiskType              * Disk Type code
    BPTR id_VolumeNode            * BCPL pointer to volume node
    LONG id_InUse                 * Flag, zero if not in use
    LABEL id_SIZEOF               * InfoData

* ID stands for InfoData
*               Disk states
ID_WRITE_PROTECTED       EQU     80      * Disk is write protected
ID_VALIDATING            EQU     81      * Disk is currently being validated
ID_VALIDATED             EQU     82      * Disk is consistent and writeable
*              Disk types
ID_NO_DISK_PRESENT       EQU -1
ID_UNREADABLE_DISK       EQU   ('B'<<24)!('A'<<16)!('D'<<8)
ID_NOT_REALLY_DOS        EQU   ('N'<<24)!('D'<<16)!('O'<<8)!('S')
ID_DOS_DISK              EQU   ('D'<<24)!('O'<<16)!('S'<<8)
ID_FFS_DISK              EQU   ('D'<<24)!('O'<<16)!('S'<<8)!(1)
```

dos/dos.i Page 3

```
ID_KICKSTART_DISK       EQU   ('K'<<24)!('I'<<16)!('C'<<8)!('K')
ID_MSDOS_DISK           EQU   ('M'<<24)!('S'<<16)!('D'<<8)

* Errors from IoErr(), etc.
ERROR_NO_FREE_STORE                  EQU  103
ERROR_TASK_TABLE_FULL                EQU  105
ERROR_BAD_TEMPLATE                   EQU  114
ERROR_BAD_NUMBER                     EQU  115
ERROR_REQUIRED_ARG_MISSING           EQU  116
ERROR_KEY_NEEDS_ARG                  EQU  117
ERROR_TOO_MANY_ARGS                  EQU  118
ERROR_UNMATCHED_QUOTES               EQU  119
ERROR_LINE_TOO_LONG                  EQU  120
ERROR_FILE_NOT_OBJECT                EQU  121
ERROR_INVALID_RESIDENT_LIBRARY       EQU  122
ERROR_NO_DEFAULT_DIR                 EQU  201
ERROR_OBJECT_IN_USE                  EQU  202
ERROR_OBJECT_EXISTS                  EQU  203
ERROR_DIR_NOT_FOUND                  EQU  204
ERROR_OBJECT_NOT_FOUND               EQU  205
ERROR_BAD_STREAM_NAME                EQU  206
ERROR_OBJECT_TOO_LARGE               EQU  207
ERROR_ACTION_NOT_KNOWN               EQU  209
ERROR_INVALID_COMPONENT_NAME         EQU  210
ERROR_INVALID_LOCK                   EQU  211
ERROR_OBJECT_WRONG_TYPE              EQU  212
ERROR_DISK_NOT_VALIDATED             EQU  213
ERROR_DISK_WRITE_PROTECTED           EQU  214
ERROR_RENAME_ACROSS_DEVICES          EQU  215
ERROR_DIRECTORY_NOT_EMPTY            EQU  216
ERROR_TOO_MANY_LEVELS                EQU  217
ERROR_DEVICE_NOT_MOUNTED             EQU  218
ERROR_SEEK_ERROR                     EQU  219
ERROR_COMMENT_TOO_BIG                EQU  220
ERROR_DISK_FULL                      EQU  221
ERROR_DELETE_PROTECTED               EQU  222
ERROR_WRITE_PROTECTED                EQU  223
ERROR_READ_PROTECTED                 EQU  224
ERROR_NOT_A_DOS_DISK                 EQU  225
ERROR_NO_DISK                        EQU  226
ERROR_NO_MORE_ENTRIES                EQU  232
* added for 1.4
ERROR_IS_SOFT_LINK                   EQU  233
ERROR_OBJECT_LINKED                  EQU  234
ERROR_BAD_HUNK                       EQU  235
ERROR_NOT_IMPLEMENTED                EQU  236
ERROR_RECORD_NOT_LOCKED              EQU  240
ERROR_LOCK_COLLISION                 EQU  241
ERROR_LOCK_TIMEOUT                   EQU  242
ERROR_UNLOCK_ERROR                   EQU  243

* error codes 303-305 are defined in dosasl.i

* These are the return codes used by convention by AmigaDOS commands
* See FAILAT and IF for relvance to EXECUTE files
RETURN_OK                            EQU    0  * No problems, success
RETURN_WARN                          EQU    5  * A warning only
RETURN_ERROR                         EQU   10  * Something wrong
RETURN_FAIL                          EQU   20  * Complete or severe failure

* Bit numbers that signal you that a user has issued a break
* for example: if (SetSignal(0,0) & SIGBREAKF_CTRL_C) cleanup_and_exit();
        BITDEF  SIGBREAK,CTRL_C,12
        BITDEF  SIGBREAK,CTRL_D,13
        BITDEF  SIGBREAK,CTRL_E,14
        BITDEF  SIGBREAK,CTRL_F,15
```

dos/dos.i

Page 4

```

* Values returned by SameLock()
LOCK_SAME               EQU     0
LOCK_SAME_HANDLER       EQU     1
LOCK_DIFFERENT          EQU     -1

* types for ChangeMode()
CHANGE_LOCK     EQU     0
CHANGE_FH       EQU     1

* Values for MakeLink()
LINK_HARD       EQU     0
LINK_SOFT       EQU     1               ; softlinks are not fully supported yet

* values returned by ReadItem
ITEM_EQUAL      EQU     -2              ; "=" Symbol
ITEM_ERROR      EQU     -1              ; error
ITEM_NOTHING    EQU     0               ; *N, ;, endstreamch
ITEM_UNQUOTED   EQU     1               ; unquoted item
ITEM_QUOTED     EQU     2               ; quoted item

* types for AllocDosObject/FreeDosObject
DOS_FILEHANDLE          EQU     0       ; few people should use this
DOS_EXALLCONTROL        EQU     1       ; Must be used to allocate this!
DOS_FIB                 EQU     2       ; useful
DOS_STDPKT              EQU     3       ; for doing packet-level I/O
DOS_CLI                 EQU     4       ; for shell-writers, etc
DOS_RDARGS              EQU     5       ; for ReadArgs if you pass it in

        ENDC    ; DOS_DOS_I
```

dos/dos_lib.i

Page 1

```
        IFND    DOS_DOS_LIB_I
DOS_DOS_LIB_I   SET     1
**
**      $Filename: dos/dos_lib.i $
**      $Release: 2.04 $
**      $Revision: 36.1 $
**      $Date: 90/11/04 $
**
**      Library interface offsets for DOS library
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

reserve EQU     4
vsize   EQU     6
count   SET     -vsize*(reserve+1)
LIBENT  MACRO
_LVO\1  EQU     count
count   SET     count-vsize
        ENDM
*
*
*
   LIBENT   Open
   LIBENT   Close
   LIBENT   Read
   LIBENT   Write
   LIBENT   Input
   LIBENT   Output
   LIBENT   Seek
   LIBENT   DeleteFile
   LIBENT   Rename
   LIBENT   Lock
   LIBENT   UnLock
   LIBENT   DupLock
   LIBENT   Examine
   LIBENT   ExNext
   LIBENT   Info
   LIBENT   CreateDir
   LIBENT   CurrentDir
   LIBENT   IoErr
   LIBENT   CreateProc
   LIBENT   Exit
   LIBENT   LoadSeg
   LIBENT   UnLoadSeg
   LIBENT   GetPacket
   LIBENT   QueuePacket
   LIBENT   DeviceProc
   LIBENT   SetComment
   LIBENT   SetProtection
   LIBENT   DateStamp
   LIBENT   Delay
   LIBENT   WaitForChar
   LIBENT   ParentDir
   LIBENT   IsInteractive
   LIBENT   Execute

        ENDC    ; DOS_DOS_LIB_I
        IFND DOS_DOSASL_I
DOS_DOSASL_I    SET     1
**
**      $Filename: dos/dosasl.i $
**      $Release: 2.04 $
**      $Revision: 36.18 $
**      $Date: 91/01/27 $
**
**      pattern-matching structure definitions
**
**      (C) Copyright 1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**

        IFND EXEC_LIBRARIES_I
        INCLUDE "exec/libraries.i"
        ENDC

        IFND EXEC_LISTS_I
        INCLUDE "exec/lists.i"
        ENDC

        IFND DOS_DOS_I
        INCLUDE "dos/dos.i"
        ENDC

************************************************************************
************************ PATTERN MATCHING ******************************
************************************************************************

* structure expected by MatchFirst, MatchNext.
* Allocate this structure and initialize it as follows:
*
* Set ap_BreakBits to the signal bits (CDEF) that you want to take a
* break on, or NULL, if you don't want to convenience the user.
*
* If you want to have the FULL PATH NAME of the files you found,
* allocate a buffer at the END of this structure, and put the size of
* it into ap_Strlen.  If you don't want the full path name, make sure
* you set ap_Strlen to zero.  In this case, the name of the file, and stats
* are available in the ap_Info, as per usual.
*
* Then call MatchFirst() and then afterwards, MatchNext() with this structure.
* You should check the return value each time (see below) and take the
* appropriate action, ultimately calling MatchEnd() when there are
* no more files and you are done.  You can tell when you are done by
* checking for the normal AmigaDOS return code ERROR_NO_MORE_ENTRIES.
*


        STRUCTURE AnchorPath,0
                LABEL   ap_First
                CPTR    ap_Base ; pointer to first anchor
                LABEL   ap_Current
                CPTR    ap_Last ; pointer to last anchor
                LONG    ap_BreakBits    ; Bits we want to break on
                LONG    ap_FoundBreak   ; Bits we broke on. Also returns ERROR_B
  REAK
                LABEL   ap_Length       ; Old compatability for LONGWORD ap_Leng
  th
                BYTE    ap_Flags        ; New use for extra word.
                BYTE    ap_Reserved
                WORD    ap_Strlen       ; This is what ap_Length used to be
                STRUCT  ap_Info,fib_SIZEOF      ; FileInfoBlock
                LABEL   ap_Buf          ; Buffer for path name, allocated by use
  r
                LABEL   ap_SIZEOF
```

```


        BITDEF  AP,DOWILD,0                 ; User option ALL
        BITDEF  AP,ITSWILD,1                ; Set by MatchFirst, used by MatchNext
                                            ; Application can test APB_ITSWILD,
                                            ; too (means that there's a wildcard
                                            ; in the pattern after calling
                                            ; MatchFirst).
        BITDEF  AP,DODIR,2                  ; Bit is SET if a DIR node should be
                                            ; entered. Application can RESET this
                                            ; bit after MatchFirst/MatchNext to
                                            ; AVOID entering a dir.
        BITDEF  AP,DIDDIR,3                 ; Bit is SET for an "expired" dir node.
        BITDEF  AP,NOMEMERR,4               ; Set on memory error
        BITDEF  AP,DODOT,5                  ; If set, allow conversion of '.' to
                                            ; CurrentDir
        BITDEF  AP,DirChanged,6             ; ap_Current->an_Lock changed
                                            ; since last MatchNext call

        STRUCTURE       AChain,0
                CPTR    an_Child
                CPTR    an_Parent
                LONG    an_Lock
                STRUCT  an_Info,fib_SIZEOF      ; FileInfoBlock
                BYTE    an_Flags
                LABEL   an_String
                LABEL   an_SIZEOF

        BITDEF  DD,PatternBit,0
        BITDEF  DD,ExaminedBit,1
        BITDEF  DD,Completed,2
        BITDEF  DD,AllBit,3
        BITDEF  DD,SINGLE,4

* Constants used by wildcard routines, these are the pre-parsed tokens
* referred to by pattern match.  It is not necessary for you to do
* anything about these, MatchFirst() MatchNext() handle all these for you.

P_ANY           EQU     $80     ; Token for '*' or '#?
P_SINGLE        EQU     $81     ; Token for '?'
P_ORSTART       EQU     $82     ; Token for '('
P_ORNEXT        EQU     $83     ; Token for '|'
P_OREND EQU     $84     ; Token for ')'
P_NOT           EQU     $85     ; Token for '~'
P_NOTEND        EQU     $86     ; Token for
P_NOTCLASS      EQU     $87     ; Token for '^'
P_CLASS EQU     $88     ; Token for '[]'
P_REPBEG        EQU     $89     ; Token for '['
P_REPEND        EQU     $8A     ; Token for ']'
P_STOP          EQU     $8B     ; token to force end of evaluation

* Values for an_Status, NOTE: These are the actual bit numbers

COMPLEX_BIT     EQU     1       ; Parsing complex pattern
EXAMINE_BIT     EQU     2       ; Searching directory

* Returns from MatchFirst(), MatchNext()
* You can also get dos error returns, such as ERROR_NO_MORE_ENTRIES,
* these are in the dos.h file.
*
ERROR_BUFFER_OVERFLOW   EQU     303     ; User or internal buffer overflow
ERROR_BREAK             EQU     304     ; A break character was received
ERROR_NOT_EXECUTABLE    EQU     305     ; A file has E bit cleared

        ENDC    ; DOS_DOSASL_I
        IFND    DOS_DOSEXTENS_I
DOS_DOSEXTENS_I SET     1
**
**      $Filename: dos/dosextens.i $
**      $Release: 2.04 $
**      $Revision: 36.34 $
**      $Date: 91/02/25 $
**
**      DOS structures not needed for the casual AmigaDOS user
**
**      (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**

     IFND  EXEC_TYPES_I
     INCLUDE "exec/types.i"
     ENDC
     IFND  EXEC_TASKS_I
     INCLUDE "exec/tasks.i"
     ENDC
     IFND  EXEC_PORTS_I
     INCLUDE "exec/ports.i"
     ENDC
     IFND  EXEC_LIBRARIES_I
     INCLUDE "exec/libraries.i"
     ENDC
     IFND  EXEC_SEMAPHORES_I
     INCLUDE "exec/semaphores.i"
     ENDC
     IFND  DEVICES_TIMER_I
     INCLUDE "devices/timer.i"
     ENDC

     IFND  DOS_DOS_I
     INCLUDE "dos/dos.i"
     ENDC


* All DOS processes have this STRUCTure
* Create and DeviceProc returns pointer to the MsgPort in this STRUCTure
* Process_addr = DeviceProc(..) - TC_SIZE

 STRUCTURE Process,0
    STRUCT  pr_Task,TC_SIZE
    STRUCT  pr_MsgPort,MP_SIZE  * This is BPTR address from DOS functions
    WORD    pr_Pad              * Remaining variables on 4 byte boundaries
    BPTR    pr_SegList          * Array of seg lists used by this process
    LONG    pr_StackSize        * Size of process stack in bytes
    APTR    pr_GlobVec          * Global vector for this process (BCPL)
    LONG    pr_TaskNum          * CLI task number of zero if not a CLI
    BPTR    pr_StackBase        * Ptr to high memory end of process stack
    LONG    pr_Result2          * Value of secondary result from last call
    BPTR    pr_CurrentDir       * Lock associated with current directory
    BPTR    pr_CIS              * Current CLI Input Stream
    BPTR    pr_COS              * Current CLI Output Stream
    APTR    pr_ConsoleTask      * Console handler process for current window
    APTR    pr_FileSystemTask   * File handler process for current drive
    BPTR    pr_CLI              * pointer to CommandLineInterface
    APTR    pr_ReturnAddr       * pointer to previous stack frame
    APTR    pr_PktWait          * Function to be called when awaiting msg
    APTR    pr_WindowPtr        * Window pointer for errors

* following definitions are new with 2.0
    BPTR    pr_HomeDir          * Home directory of executing program
    LONG    pr_Flags            * flags telling dos about process
    APTR    pr_ExitCode         * code to call on exit of program or NULL
```

```
    LONG    pr_ExitData         * Passed as an argument to pr_ExitCode
    APTR    pr_Arguments        * Arguments passed to the process at start
    STRUCT  pr_LocalVars,MLH_SIZE * Local environment variables
    APTR    pr_ShellPrivate     * for the use of the current shell
    BPTR    pr_CES              * Error stream - if NULL, use pr_COS
   LABEL   pr_SIZEOF            * Process

*
* Flags for pr_Flags
*
        BITDEF PR,FREESEGLIST,0
        BITDEF PR,FREECURRDIR,1
        BITDEF PR,FREECLI,2
        BITDEF PR,CLOSEINPUT,3
        BITDEF PR,CLOSEOUTPUT,4
        BITDEF PR,FREEARGS,5

* The long word address (BPTR) of this STRUCTure is returned by
* Open() and other routines that return a file.  You need only worry
* about this STRUCT to do async io's via PutMsg() instead of
* standard file system calls

 STRUCTURE FileHandle,0
   APTR   fh_Link               * pointer to EXEC message
   APTR   fh_Interactive        * Boolean; TRUE if interactive handle
   APTR   fh_Type               * Port to do PutMsg() to
   LONG   fh_Buf
   LONG   fh_Pos
   LONG   fh_End
   LONG   fh_Funcs
fh_Func1 EQU fh_Funcs
   LONG   fh_Func2
   LONG   fh_Func3
   LONG   fh_Args
fh_Arg1 EQU fh_Args
   LONG   fh_Arg2
   LABEL  fh_SIZEOF * FileHandle

* This is the extension to EXEC Messages used by DOS
 STRUCTURE DosPacket,0
   APTR   dp_Link            * pointer to EXEC message
   APTR   dp_Port            * pointer to Reply port for the packet
*                            * Must be filled in each send.
   LONG   dp_Type            * See ACTION_... below and
*                            * 'R' means Read, 'W' means Write to the file system
   LONG   dp_Res1            * For file system calls this is the result
*                            * that would have been returned by the
*                            * function, e.g. Write ('W') returns actual
*                            * length written
   LONG   dp_Res2            * For file system calls this is what would
*                            * have been returned by IoErr()
   LONG   dp_Arg1
*  Device packets common equivalents
dp_Action  EQU  dp_Type
dp_Status  EQU  dp_Res1
dp_Status2 EQU  dp_Res2
dp_BufAddr EQU  dp_Arg1
   LONG   dp_Arg2
   LONG   dp_Arg3
   LONG   dp_Arg4
   LONG   dp_Arg5
   LONG   dp_Arg6
   LONG   dp_Arg7
   LABEL  dp_SIZEOF * DosPacket

* A Packet does not require the Message to before it in memory, but
* for convenience it is useful to associate the two.
* Also see the function init_std_pkt for initializing this STRUCTure

 STRUCTURE StandardPacket,0
   STRUCT sp_Msg,MN_SIZE
   STRUCT sp_Pkt,dp_SIZEOF
   LABEL  sp_SIZEOF * StandardPacket


* Packet types
ACTION_NIL              EQU     0
ACTION_STARTUP          EQU     0
ACTION_GET_BLOCK        EQU     2       ;OBSOLETE
ACTION_SET_MAP          EQU     4
ACTION_DIE              EQU     5
ACTION_EVENT            EQU     6
ACTION_CURRENT_VOLUME   EQU     7
ACTION_LOCATE_OBJECT    EQU     8
ACTION_RENAME_DISK      EQU     9
ACTION_WRITE            EQU     'W'
ACTION_READ             EQU     'R'
ACTION_FREE_LOCK        EQU     15
ACTION_DELETE_OBJECT    EQU     16
ACTION_RENAME_OBJECT    EQU     17
ACTION_MORE_CACHE       EQU     18
ACTION_COPY_DIR         EQU     19
ACTION_WAIT_CHAR        EQU     20
ACTION_SET_PROTECT      EQU     21
ACTION_CREATE_DIR       EQU     22
ACTION_EXAMINE_OBJECT   EQU     23
ACTION_EXAMINE_NEXT     EQU     24
ACTION_DISK_INFO        EQU     25
ACTION_INFO             EQU     26
ACTION_FLUSH            EQU     27
ACTION_SET_COMMENT      EQU     28
ACTION_PARENT           EQU     29
ACTION_TIMER            EQU     30
ACTION_INHIBIT          EQU     31
ACTION_DISK_TYPE        EQU     32
ACTION_DISK_CHANGE      EQU     33
ACTION_SET_DATE         EQU     34

ACTION_SCREEN_MODE      EQU     994

ACTION_READ_RETURN      EQU     1001
ACTION_WRITE_RETURN     EQU     1002
ACTION_SEEK             EQU     1008
ACTION_FINDUPDATE       EQU     1004
ACTION_FINDINPUT        EQU     1005
ACTION_FINDOUTPUT       EQU     1006
ACTION_END              EQU     1007
ACTION_SET_FILE_SIZE    EQU     1022    ; fast file system only
ACTION_WRITE_PROTECT    EQU     1023    ; fast file system only

* new 2.0 packets
ACTION_SAME_LOCK        EQU     40
ACTION_CHANGE_SIGNAL    EQU     995
ACTION_FORMAT           EQU     1020
ACTION_MAKE_LINK        EQU     1021
*
*
ACTION_READ_LINK        EQU     1024
ACTION_FH_FROM_LOCK     EQU     1026
ACTION_IS_FILESYSTEM    EQU     1027
ACTION_CHANGE_MODE      EQU     1028
*
```

```
ACTION_COPY_DIR_FH      EQU     1030
ACTION_PARENT_FH        EQU     1031
ACTION_EXAMINE_ALL      EQU     1033
ACTION_EXAMINE_FH       EQU     1034

ACTION_LOCK_RECORD      EQU     2008
ACTION_FREE_RECORD      EQU     2009

ACTION_ADD_NOTIFY       EQU     4097
ACTION_REMOVE_NOTIFY    EQU     4098

* A structure for holding error messages - stored as array with error == 0
* for the last entry.

 STRUCTURE ErrorString,0
       APTR   estr_Nums
       APTR   estr_Strings
       LABEL  ErrorString_SIZEOF

* DOS library node structure.
* This is the data at positive offsets from the library node.
* Negative offsets from the node is the jump table to DOS functions
* node = (STRUCT DosLibrary *) OpenLibrary( "dos.library" .. )

 STRUCTURE DosLibrary,0
    STRUCT  dl_lib,LIB_SIZE
    APTR    dl_Root              * Pointer to RootNode, described below
    APTR    dl_GV                * Pointer to BCPL global vector
    LONG    dl_A2                * Private register dump of DOS
    LONG    dl_A5
    LONG    dl_A6
    APTR    dl_Errors            * pointer to array of error msgs
    APTR    dl_TimeReq           * private! pointer to timer request
    APTR    dl_UtilityBase       * private! pointer to utility library base
    LABEL   dl_SIZEOF *  DosLibrary

*

 STRUCTURE RootNode,0
    BPTR    rn_TaskArray           * [0] is max number of CLI's
*                                  * [1] is APTR to process id of CLI 1
*                                  * [n] is APTR to process id of CLI n
    BPTR    rn_ConsoleSegment      * SegList for the CLI
    STRUCT  rn_Time,ds_SIZEOF      * Current time
    LONG    rn_RestartSeg          * SegList for the disk validator process
    BPTR    rn_Info                * Pointer to the Info structure
    BPTR    rn_FileHandlerSegment * code for file handler
    STRUCT  rn_CliList,MLH_SIZE * new list of all CLI processes
*                                  * the first cpl_Array is also rn_TaskArray
    APTR    rn_BootProc            * private! ptr to msgport of boot fs
    BPTR    rn_ShellSegment        * seglist for Shell (for NewShell)
    LONG    rn_Flags               * dos flags
    LABEL   rn_SIZEOF * RootNode

 BITDEF RN,WILDSTAR,24
 BITDEF RN,PRIVATE1,1

* ONLY to be allocated by DOS!
 STRUCTURE CliProcList,0
    STRUCT  cpl_Node,MLN_SIZE
    LONG    cpl_First              * number of first entry in array
    APTR    cpl_Array              * pointer to array of process msgport pointers
*                                  * [0] is max number of CLI's in this entry (n)
*                                  * [1] is APTR to process id of CLI cpl_First
*                                  * [n] is APTR to process id of CLI cpl_First+n-1
    LABEL   cpl_SIZEOF

 STRUCTURE DosInfo,0
    BPTR    di_McName            * PRIVATE: system resident module list
di_ResList EQU di_McName
    BPTR    di_DevInfo           * Device List
    BPTR    di_Devices           * Currently zero
    BPTR    di_Handlers          * Currently zero
    APTR    di_NetHand           * Network handler processid currently zero
    STRUCT  di_DevLock,SS_SIZE     * do NOT access directly!
    STRUCT  di_EntryLock,SS_SIZE  * do NOT access directly!
    STRUCT  di_DeleteLock,SS_SIZE * do NOT access directly!
    LABEL   di_SIZEOF * DosInfo

* structure for the Dos resident list.  Do NOT allocate these, use
* AddSegment(), and heed the warnings in the autodocs!

 STRUCTURE Segment,0
        BPTR seg_Next
        LONG seg_UC
        BPTR seg_Seg
        STRUCT seg_Name,4       ; Actually the first 4 chars of BSTR name
        LABEL seg_SIZEOF

CMD_SYSTEM      EQU     -1
CMD_INTERNAL    EQU     -2
CMD_DISABLED    EQU     -999


* DOS Processes started from the CLI via RUN or NEWCLI have this additional
* set to data associated with them

 STRUCTURE CommandLineInterface,0
    LONG   cli_Result2          * Value of IoErr from last command
    BSTR   cli_SetName          * Name of current directory
    BPTR   cli_CommandDir       * Head of the path locklist
    LONG   cli_ReturnCode       * Return code from last command
    BSTR   cli_CommandName      * Name of current command
    LONG   cli_FailLevel        * Fail level (set by FAILAT)
    BSTR   cli_Prompt           * Current prompt (set by PROMPT)
    BPTR   cli_StandardInput    * Default (terminal) CLI input
    BPTR   cli_CurrentInput     * Current CLI input
    BSTR   cli_CommandFile      * Name of EXECUTE command file
    LONG   cli_Interactive      * Boolean True if prompts required
    LONG   cli_Background       * Boolean True if CLI created by RUN
    BPTR   cli_CurrentOutput    * Current CLI output
    LONG   cli_DefaultStack     * Stack size to be obtained in long words
    BPTR   cli_StandardOutput   * Default (terminal) CLI output
    BPTR   cli_Module           * SegList of currently loaded command
    LABEL  cli_SIZEOF           * CommandLineInterface

* This structure can take on different values depending on whether it is
* a device, an assigned directory, or a volume.  Below is the structure
* reflecting volumes only.  Following that is the structure representing
* only devices. Following that is the unioned structure representing all
* the values

* structure representing a volume

 STRUCTURE DevList,0
    BPTR        dl_Next                  ; bptr to next device list
    LONG        dl_Type                  ; see DLT below
    APTR        dl_Task                  ; ptr to handler task
    BPTR        dl_Lock                  ; not for volumes
    STRUCT      dl_VolumeDate,ds_SIZEOF ; creation date
    BPTR        dl_LockList              ; outstanding locks
    LONG        dl_DiskType              ; 'DOS', etc
```

```
    LONG        dl_unused
    BSTR        dl_Name                  ; bptr to bcpl name
    LABEL       DevList_SIZEOF

* device structure (same as the DeviceNode structure in filehandler.i

 STRUCTURE       DevInfo,0
    BPTR    dvi_Next
    LONG    dvi_Type
    APTR    dvi_Task
    BPTR    dvi_Lock
    BSTR    dvi_Handler
    LONG    dvi_Stacksize
    LONG    dvi_Priority
    LONG    dvi_Startup
    BPTR    dvi_SegList
    BPTR    dvi_GlobVec
    BSTR    dvi_Name
    LABEL   dvi_SIZEOF

* combined structure for devices, assigned directories, volumes

 STRUCTURE DosList,0
    BPTR     dol_Next        ; bptr to next device on lis
    LONG     dol_Type        ; see DLT below
    APTR     dol_Task        ; ptr to handler task
    BPTR     dol_Lock

    STRUCT      dol_VolumeDate,0        ; creation date (UNION)
    STRUCT      dol_AssignName,0        ; name for assign path (UNION)
    BSTR        dol_Handler             ; file name to load if seglist is null
    STRUCT      dol_List,0              ; List of directories assigned (UNION)
    LONG        dol_StackSize           ; stacksize to use when starting process
    LONG        dol_Priority            ; task priority when starting process

    STRUCT      dol_LockList,0          ; outstanding locks (UNION)
    ULONG       dol_Startup             ; startup msg: FileSysStartupMsg
                                        ; for disks

    STRUCT      dol_DiskType,0          ; 'DOS', etc (UNION)
    BPTR        dol_SegList             ; already loaded code for new task

    BPTR        dol_GlobVec             ; BCPL global vector

    BSTR        dol_Name                ; bptr to bcpl name
    LABEL       DosList_SIZEOF



* definitions for dl_Type
DLT_DEVICE      EQU     0
DLT_DIRECTORY   EQU     1       ; assign
DLT_VOLUME      EQU     2
DLT_LATE        EQU     3       ; late-binding assign
DLT_NONBINDING  EQU     4       ; non-binding assign (AssignPath)
DLT_PRIVATE     EQU     -1      ; for internal use only

* structure return by GetDeviceProc()
 STRUCTURE DevProc,0
    APTR        dvp_Port        ; struct MsgPort *
    BPTR        dvp_Lock
    ULONG       dvp_Flags
    APTR        dvp_DevNode     ; struct DosList * - DON'T TOUCH OR USE!
    LABEL       dvp_SIZEOF

* definitions for dvp_Flags
```

dos/dosextens.i

```
 BITDEF DVP,UNLOCK,0
 BITDEF DVP,ASSIGN,1

* Flags to be passed to LockDosList(), etc
 BITDEF LD,DEVICES,2
 BITDEF LD,VOLUMES,3
 BITDEF LD,ASSIGNS,4
 BITDEF LD,ENTRY,5
 BITDEF LD,DELETE,6

* You MUST specify one of LDF_READ or LDF_WRITE
 BITDEF LD,READ,0
 BITDEF LD,WRITE,1

* actually all but LDF_ENTRY (which is used for internal locking)
LDF_ALL EQU     (LDF_DEVICES!LDF_VOLUMES!LDF_ASSIGNS)

* a lock structure, as returned by Lock() or DupLock()
 STRUCTURE FileLock,0
    BPTR        fl_Link                 ; bcpl pointer to next lock
    LONG        fl_Key                  ; disk block number
    LONG        fl_Access               ; exclusive or shared
    APTR        fl_Task                 ; handler task's port
    BPTR        fl_Volume               ; bptr to DLT_VOLUME DosList entry
    LABEL       fl_SIZEOF

* error report types for ErrorReport()
REPORT_STREAM   EQU     0       ; a stream
REPORT_TASK     EQU     1       ; a process - unused
REPORT_LOCK     EQU     2       ; a lock
REPORT_VOLUME   EQU     3       ; a volume node
REPORT_INSERT   EQU     4       ; please insert volume

* Special error codes for ErrorReport()
ABORT_DISK_ERROR        EQU     296     ; Read/write error
ABORT_BUSY              EQU     288     ; You MUST replace...

* types for initial packets to shells from run/newcli/execute/system.
* For shell-writers only
RUN_EXECUTE             EQU     -1
RUN_SYSTEM              EQU     -2
RUN_SYSTEM_ASYNCH       EQU     -3

* Types for fib_DirEntryType.  NOTE that both USERDIR and ROOT are
* directories, and that directory/file checks should use <0 and >=0.
* This is not necessarily exhaustive!  Some handlers may use other
* values as needed, though <0 and >=0 should remain as supported as
* possible.
ST_ROOT         EQU     1
ST_USERDIR      EQU     2
ST_SOFTLINK     EQU     3       ; looks like dir, but may point to a file!
ST_LINKDIR      EQU     4       ; hard link to dir
ST_FILE         EQU     -3      ; must be negative for FIB!
ST_LINKFILE     EQU     -4      ; hard link to file

        ENDC    ; DOS_DOSEXTENS_I
```

dos/doshunks.i

```
        IFND    DOS_DOSHUNKS_I
DOS_DOSHUNKS_I  SET     1
**
**      $Filename: dos/doshunks.i $
**      $Release: 2.04 $
**      $Revision: 36.6 $
**      $Date: 91/02/10 $
**
**      Hunk definitions for object and load modules.
**
**      (C) Copyright 1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**

* hunk types

HUNK_UNIT       EQU     999
HUNK_NAME       EQU     1000
HUNK_CODE       EQU     1001
HUNK_DATA       EQU     1002
HUNK_BSS        EQU     1003
HUNK_RELOC32    EQU     1004
HUNK_RELOC16    EQU     1005
HUNK_RELOC8     EQU     1006
HUNK_EXT        EQU     1007
HUNK_SYMBOL     EQU     1008
HUNK_DEBUG      EQU     1009
HUNK_END        EQU     1010
HUNK_HEADER     EQU     1011

HUNK_OVERLAY    EQU     1013
HUNK_BREAK      EQU     1014

HUNK_DREL32     EQU     1015
HUNK_DREL16     EQU     1016
HUNK_DREL8      EQU     1017

HUNK_LIB        EQU     1018
HUNK_INDEX      EQU     1019

* hunk_ext sub-types

EXT_SYMB        EQU     0       ; symbol table
EXT_DEF         EQU     1       ; relocatable definition
EXT_ABS         EQU     2       ; Absolute definition
EXT_RES         EQU     3       ; no longer supported
EXT_REF32       EQU     129     ; 32 bit reference to symbol
EXT_COMMON      EQU     130     ; 32 bit reference to COMMON block
EXT_REF16       EQU     131     ; 16 bit reference to symbol
EXT_REF8        EQU     132     ;  8 bit reference to symbol
EXT_DEXT32      EQU     133     ; 32 bit data releative reference
EXT_DEXT16      EQU     134     ; 16 bit data releative reference
EXT_DEXT8       EQU     135     ;  8 bit data releative reference

        ENDC    ; DOS_DOSHUNKS_I
```

dos/dostags.i

```
        IFND    DOS_DOSTAGS_I
DOS_DOSTAGS_I SET       1
**
**      $Filename: dos/dostags.i $
**      $Release: 2.04 $
**      $Revision: 36.11 $
**      $Date: 90/11/02 $
**
**      Tag definitions for all Dos routines using tags
**
**      (C) Copyright 1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**


     IFND  UTILITY_TAGITEM_I
     INCLUDE "utility/tagitem.i"
     ENDC

*****************************************************************************
* definitions for the System() call

SYS_Dummy       EQU     TAG_USER+32
SYS_Input       EQU     SYS_Dummy+1     ; specifies the input filehandle
SYS_Output      EQU     SYS_Dummy+2     ; specifies the output filehandle
SYS_Asynch      EQU     SYS_Dummy+3     ; run asynch, close input/output(!)
SYS_UserShell   EQU     SYS_Dummy+4   ; send to user shell instead of boot shell
SYS_CustomShell EQU     SYS_Dummy+5   ; send to a specific shell (data is name)
*SYS_Error      EQU     SYS_Dummy+?


*****************************************************************************
* definitions for the CreateNewProc() call
* you MUST specify one of NP_Seglist or NP_Entry.  All else is optional.

NP_Dummy        EQU TAG_USER+1000
NP_Seglist      EQU NP_Dummy+1   ; seglist of code to run for the process
NP_FreeSeglist  EQU NP_Dummy+2   ; free seglist on exit - only valid for
                                 ; for NP_Seglist.  Default is TRUE.
NP_Entry        EQU NP_Dummy+3   ; entry point to run - mutually exclusive
                                 ; with NP_Seglist.
NP_Input        EQU NP_Dummy+4   ; filehandle - default is Open("NIL:"...)
NP_Output       EQU NP_Dummy+5   ; filehandle - default is Open("NIL:"...)
NP_CloseInput   EQU NP_Dummy+6   ; close input filehandle on exit
                                 ; default TRUE
NP_CloseOutput  EQU NP_Dummy+7   ; close output filehandle on exit
                                 ; default TRUE
NP_Error        EQU NP_Dummy+8   ; filehandle - default is Open("NIL:"...)
NP_CloseError   EQU NP_Dummy+9   ; close error filehandle on exit
                                 ; default TRUE
NP_CurrentDir   EQU NP_Dummy+10  ; lock - default is parent's current dir
NP_StackSize    EQU NP_Dummy+11  ; stacksize for process - default 4000
NP_Name         EQU NP_Dummy+12  ; name for process - default "New Process"
NP_Priority     EQU NP_Dummy+13  ; priority - default same as parent
NP_ConsoleTask  EQU NP_Dummy+14  ; consoletask - default same as parent
NP_WindowPtr    EQU NP_Dummy+15  ; window ptr - default is same as parent
NP_HomeDir      EQU NP_Dummy+16  ; home directory - default current home dir
NP_CopyVars     EQU NP_Dummy+17  ; boolean to copy local vars-default TRUE
NP_Cli          EQU NP_Dummy+18  ; create cli structure - default FALSE
NP_Path         EQU NP_Dummy+19  ; path - default is copy of parents path
                                 ; only valid if a cli process!
NP_CommandName  EQU NP_Dummy+20  ; commandname - valid only for CLI
NP_Arguments    EQU NP_Dummy+21  ; cstring of arguments - passed with str
                                 ; in a0, length in d0.  (copied and freed
                                 ; on exit.  Default is empty string.
                                 ; NOTE: not operational until 2.04 - see
```

dos/dostags.i

```
                                 ; BIX/TechNotes for more info/workarounds
                                 ; NOTE: in 2.0, it DIDN'T pass "" - the
                                 ; registers were random.
NP_NotifyOnDeath EQU NP_Dummy+22 ; notify parent on death - default FALSE
                                 ; Not functional yet.
NP_Synchronous   EQU NP_Dummy+23 ; don't return until process finishes -
                                 ; default FALSE.
                                 ; Not functional yet.
NP_ExitCode      EQU NP_Dummy+24 ; code to be called on process exit
NP_ExitData      EQU NP_Dummy+25 ; optional argument for NP_EndCode rtn -
                                 ; default NULL

*****************************************************************************
* tags for AllocDosObject
ADO_Dummy        EQU      TAG_USER+2000
ADO_FH_Mode      EQU      ADO_Dummy+1     ; for type DOS_FILEHANDLE only
                                ; sets up FH for the type of open being done
                                ; This can make a big difference for buffered
                                ; files.

        ; The following are for DOS_CLI
        ; If you do not specify these, dos will use it's preferred values
        ; which may change from release to release.  The BPTRs to these
        ; will be set up correctly for you.  Everything will be zero,
        ; except cli_FailLevel (10) and cli_Background (DOSTRUE).
        ; NOTE: you may also use these 4 tags with CreateNewProc.

ADO_DirLen       EQU      ADO_Dummy+2      ; size in bytes for current dir buffer
ADO_CommNameLen  EQU      ADO_Dummy+3      ; size in bytes for command name buffer
ADO_CommFileLen  EQU      ADO_Dummy+4      ; size in bytes for command file buffer
ADO_PromptLen    EQU      ADO_Dummy+5      ; size in bytes for the prompt buffer

* tags for NewLoadSeg
* no tags are defined yet for NewLoadSeg

        ENDC    ; DOS_DOSTAGS_I
```

```
          IFND DOS_EXALL_I
DOS_EXALL_I SET 1
**
**        $Filename: dos/exall.i $
**        $Release: 2.04 $
**        $Revision: 36.5 $
**        $Date: 90/07/12 $
**
**        include file for ExAll() data structures
**
**        (C) Copyright 1989,1990 Commodore-Amiga, Inc.
**            All Rights Reserved
**

     IFND  EXEC_TYPES_I
     INCLUDE "exec/types.i"
     ENDC

     IFND UTILITY_HOOKS_I
     INCLUDE "utility/hooks.i"
     ENDC

* values that can be passed for what data you want from ExAll()
* each higher value includes those below it (numerically)
* you MUST chose one of these values

ED_NAME         EQU     1
ED_TYPE         EQU     2
ED_SIZE         EQU     3
ED_PROTECTION   EQU     4
ED_DATE         EQU     5
ED_COMMENT      EQU     6

*
*    Structure in which exall results are returned in.  Note that only the
*    fields asked for will exist!
*

 STRUCTURE ExAllData,0
        APTR    ed_Next         ; next struct ExAllData
        APTR    ed_Name         ; to CSTR
        LONG    ed_Type         ; type of file/dir
        ULONG   ed_Size         ; size of file (if file) in bytes
        ULONG   ed_Prot         ; protection bits
        ULONG   ed_Days         ; datestamp - last modification
        ULONG   ed_Mins
        ULONG   ed_Ticks
        APTR    ed_Comment      ; strings will be after last used field
        LABEL   ed_Strings      ; strings will start after the last USED field
 LABEL ExAllData_SIZEOF

*
*    Control structure passed to ExAll.  Unused fields MUST be initialized to
*    0, expecially eac_LastKey.
*
*    eac_MatchFunc is a hook (see utility.library documentation for usage)
*    It should return true if the entry is to returned, false if it is to be
*    ignored.
*
*    This structure MUST be allocated by AllocDosObject()!
*

 STRUCTURE ExAllControl,0
        ULONG   eac_Entries      ; number of entries returned in buffer
        ULONG   eac_LastKey      ; Don't touch inbetween linked ExAll calls!
        APTR    eac_MatchString  ; wildcard CSTR for pattern match or NULL
```

```
        APTR    eac_MatchFunc    ; optional private wildcard function hook
 LABEL ExAllControl_SIZEOF

        ENDC    ; DOS_EXALL_I
        IFND    DOS_FILEHANDLER_I
DOS_FILEHANDLER_I SET   1
**
**      $Filename: dos/filehandler.i $
**      $Release: 2.04 $
**      $Revision: 36.4 $
**      $Date: 90/07/12 $
**
**      device and file handler specific code for AmigaDOS
**
**      (C) Copyright 1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**

        IFND     EXEC_TYPES_I
        INCLUDE "exec/types.i"
        ENDC    ; EXEC_TYPES_I

        IFND     EXEC_PORTS_I
        INCLUDE "exec/ports.i"
        ENDC    ; EXEC_PORTS_I

        IFND     DOS_DOS_I
        INCLUDE "libraries/dos.i"
        ENDC    ; DOS_DOS_I


* The disk "environment" is a longword array that describes the
* disk geometry.  It is variable sized, with the length at the beginning.
* Here are the constants for a standard geometry.



 STRUCTURE DosEnvec,0
    ULONG de_TableSize       ; Size of Environment vector
    ULONG de_SizeBlock       ; in longwords: standard value is 128
    ULONG de_SecOrg          ; not used; must be 0
    ULONG de_Surfaces        ; # of heads (surfaces). drive specific
    ULONG de_SectorPerBlock  ; not used; must be 1
    ULONG de_BlocksPerTrack  ; blocks per track. drive specific
    ULONG de_Reserved        ; DOS reserved blocks at start of partition.
    ULONG de_PreAlloc        ; DOS reserved blocks at end of partition
    ULONG de_Interleave      ; usually 0
    ULONG de_LowCyl          ; starting cylinder. typically 0
    ULONG de_HighCyl         ; max cylinder. drive specific
    ULONG de_NumBuffers      ; Initial # DOS of buffers.
    ULONG de_BufMemType      ; type of mem to allocate for buffers
    ULONG de_MaxTransfer     ; Max number of bytes to transfer at a time
    ULONG de_Mask            ; Address Mask to block out certain memory
    LONG  de_BootPri         ; Boot priority for autoboot
    ULONG de_DosType         ; ASCII (HEX) string showing filesystem type;
                             ; 0X444F5300 is old filesystem,
                             ; 0X444F5301 is fast file system
    ULONG de_Baud            ; Baud rate for serial handler
    ULONG de_Control         ; Control word for handler/filesystem
    ULONG de_BootBlocks      ; Number of blocks containing boot code

    LABEL DosEnvec_SIZEOF

* these are the offsets into the array

DE_TABLESIZE    EQU     0       ; standard value is 11
DE_SIZEBLOCK    EQU     1       ; in longwords: standard value is 128
DE_SECORG       EQU     2       ; not used; must be 0
DE_NUMHEADS     EQU     3       ; # of heads (surfaces). drive specific
DE_SECSPERBLK   EQU     4       ; not used; must be 1
```

```
DE_BLKSPERTRACK EQU     5       ; blocks per track. drive specific
DE_RESERVEDBLKS EQU     6       ; unavailable blocks at start.  usually 2
DE_PREFAC       EQU     7       ; not used; must be 0
DE_INTERLEAVE   EQU     8       ; usually 0
DE_LOWCYL       EQU     9       ; starting cylinder. typically 0
DE_UPPERCYL     EQU     10      ; max cylinder.  drive specific
DE_NUMBUFFERS   EQU     11      ; starting # of buffers.  typically 5
DE_MEMBUFTYPE   EQU     12      ; type of mem to allocate for buffers.
DE_BUFMEMTYPE   EQU     12      ; same as above, better name
                                ; 1 is public, 3 is chip, 5 is fast
DE_MAXTRANSFER  EQU     13      ; Maximum number of bytes to transfer at a time
DE_MASK         EQU     14      ; Address Mask to block out certain memory
DE_BOOTPRI      EQU     15      ; Boot priority for autoboot
DE_DOSTYPE      EQU     16      ; ASCII (HEX) string showing filesystem type
                                ; 0X444F5300 is old filesystem,
                                ; 0X444F5301 is fast file system
DE_BAUD EQU     17      ; Baud rate for serial handler
DE_CONTROL      EQU     18      ; Control word for handler/filesystem
DE_BOOTBLOCKS   EQU     19      ; Number of blocks containing boot code

*
* The file system startup message is linked into a device node's startup
* field.  It contains a pointer to the above environment, plus the
* information needed to do an exec OpenDevice().
*

 STRUCTURE FileSysStartupMsg,0
    ULONG       fssm_Unit       ; exec unit number for this device
    BSTR        fssm_Device     ; null terminated bstring to the device name
    BPTR        fssm_Environ    ; ptr to environment table (see above)
    ULONG       fssm_Flags      ; flags for OpenDevice()
    LABEL       FileSysStartupMsg_SIZEOF


* The include file "libraries/dosextens.h" has a DeviceList structure.
* The "device list" can have one of three different things linked onto
* it.  Dosextens defines the structure for a volume.  DLT_DIRECTORY
* is for an assigned directory.  The following structure is for
* a dos "device" (DLT_DEVICE).

 STRUCTURE DeviceNode,0
    BPTR        dn_Next         ; singly linked list
    ULONG       dn_Type         ; always 0 for dos "devices"
    CPTR        dn_Task         ; standard dos "task" field.  If this is
                                ;     null when the node is accesses, a task
                                ;     will be started up
    BPTR        dn_Lock         ; not used for devices -- leave null
    BSTR        dn_Handler      ; filename to loadseg (if seglist is null)
    ULONG       dn_StackSize    ; stacksize to use when starting task
    LONG        dn_Priority     ; task priority when starting task
    BPTR        dn_Startup      ; startup msg: FileSysStartupMsg for disks
    BPTR        dn_SegList      ; code to run to start new task (if necessary).
                                ;     if null then dn_Handler will be loaded.
    BPTR        dn_GlobalVec    ; BCPL global vector to use when starting
                                ;     a task.  -1 means that dn_SegList is not
                                ;     for a bcpl program, so the dos won't
                                ;     try and construct one.  0 tell the
                                ;     dos that you obey BCPL linkage rules,
                                ;     and that it should construct a global
                                ;     vector for you.
    BSTR        dn_Name         ; the node name, e.g. '\3','D','F','3'
    LABEL       DeviceNode_SIZEOF

        ENDC    ; DOS_FILEHANDLER_I
```

dos/notify.i

Page 1

```
        IFND    DOS_NOTIFY_I
DOS_NOTIFY_I    SET     1
**
**      $Filename: dos/notify.i $
**      $Release: 2.04 $
**      $Revision: 36.5 $
**      $Date: 90/08/29 $
**
**      dos notification definitions
**
**      (C) Copyright 1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**

        IFND    EXEC_PORTS_I
        INCLUDE "exec/ports.i"
        ENDC
        IFND    EXEC_TASKS_I
        INCLUDE "exec/tasks.i"
        ENDC

* use of Class and code is discouraged for the time being - we might want to
* change things
*------ NotifyMessage Class -----------------------------------------
NOTIFY_CLASS    EQU     $40000000

*------ NotifyMessage Code -------------------------------------------
NOTIFY_CODE     EQU     $1234


* Sent to the application if SEND_MESSAGE is specified.

 STRUCTURE NotifyMessage,0
    STRUCT nm_ExecMessage,MN_SIZE
    ULONG  nm_Class
    UWORD  nm_Code
    APTR   nm_NReq                      ; don't modify the request!
    ULONG  nm_DoNotTouch                ; like it says!  For use by handlers
    ULONG  nm_DoNotTouch2               ; ditto
    LABEL  NotifyMessage_SIZEOF

* Do not modify or reuse the notifyrequest while it is active.

 STRUCTURE NotifyRequest,0
    CPTR   nr_Name                      ; Application sets this
    CPTR   nr_FullName                  ; fully expanded name created by Dos
                                        ; (handlers need nr_FullName)
    ULONG  nr_UserData                  ; for the application's use
    ULONG  nr_Flags

    ;-- nr_Msg:
    APTR   nr_Port                      ; struct MsgPort * for SEND_MESSAGE

    ;-- nr_Signal:
nr_Task EQU nr_Port                     ; struct Task * for SEND_SIGNAL
    UBYTE  nr_SignalNum                 ; for SEND_SIGNAL
    STRUCT nr_pad,3

    ;-- Reserved fields:
    STRUCT nr_Reserved,4*4

    ;-- internal for use by handlers/dos:
    ULONG  nr_MsgCount                  ; # of outstanding msgs
    APTR   nr_Handler                   ; handler sent to (for EndNotify)
    LABEL  NotifyRequest_SIZEOF

```

dos/notify.i

Page 2

```

;------ NotifyRequest Flags -----------------------------------------
        BITDEF  NR,SEND_MESSAGE,0
        BITDEF  NR,SEND_SIGNAL,1
        BITDEF  NR,WAIT_REPLY,3
        BITDEF  NR,NOTIFY_INITIAL,4

* do NOT set or remove NRF_MAGIC!  Only for use by handlers!
        BITDEF  NR,MAGIC,31

* Flags reserved for private use by the handler:
NR_HANDLER_FLAGS        EQU     $ffff0000

        ENDC    ; DOS_NOTIFY_I

        IFND    DOS_RDARGS_I
DOS_RDARGS_I SET 1
**
**      $Filename: dos/rdargs.i $
**      $Release: 2.04 $
**      $Revision: 36.7 $
**      $Date: 90/07/12 $
**
**      ReadArgs() structure definitions
**
**      (C) Copyright 1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**

        IFND EXEC_NODES_I
        INCLUDE "exec/nodes.i"
        ENDC

**********************************************************************
*
* The CSource data structure defines the input source for "RdItem()"
* as well as the ReadArgs call.  It is a publicly defined structure
* which may be used by applications which use code that follows the
* conventions defined for access.
*
* When passed to the dos.library functions, the value passed as
* struct *CSource is defined as follows:
*       if ( CSource == 0)      Use buffered IO "ReadChar()" as data source
*       else                    Use CSource for input character stream
*
* The following two pseudo-code routines define how the CSource structure
* is used:
*
* long CS_ReadChar( struct CSource *CSource )
* {
*       if ( CSource == 0 )     return ReadChar();
*       if ( CSource->CurChr >= CSource->Length )       return ENDSTREAMCHAR;
*       return CSource->Buffer[ CSource->CurChr++ ];
* }
*
* BOOL CS_UnReadChar( struct CSource *CSource )
* {
*       if ( CSource == 0 )     return UnReadChar();
*       if ( CSource->CurChr <= 0 )     return FALSE;
*       CSource->CurChr--;
*       return TRUE;
* }
*
* To initialize a struct CSource, you set CSource->CS_Buffer to
* a string which is used as the data source, and set CS_Length to
* the number of characters in the string.  Normally CS_CurChr should
* be initialized to ZERO, or left as it was from prior use as
* a CSource.
*
**********************************************************************

        STRUCTURE       CSource,0
                APTR    CS_Buffer
                LONG    CS_Length
                LONG    CS_CurChr
                LABEL   CS_SIZEOF

**********************************************************************
*
* The RDArgs data structure is the input parameter passed to the DOS
* ReadArgs() function call.
```

```
*
* The RDA_Source structure is a CSource as defined above;
* if RDA_Source.CS_Buffer is non-null, RDA_Source is used as the input
* character stream to parse, else the input comes from the buffered STDIN
* calls ReadChar/UnReadChar.
*
* RDA_DAList is a private address which is used internally to track
* allocations which are freed by FreeArgs().  This MUST be initialized
* to NULL prior to the first call to ReadArgs().
*
* The RDA_Buffer and RDA_BufSiz fields allow the application to supply
* a fixed-size buffer in which to store the parsed data.  This allows
* the application to pre-allocate a buffer rather than requiring buffer
* space to be allocated.  If either RDA_Buffer or RDA_BufSiz is NULL,
* the application has not supplied a buffer.
*
* RDA_ExtHelp is a text string which will be displayed instead of the
* template string, if the user is prompted for input.
*
* RDA_Flags bits control how ReadArgs() works.  The flag bits are
* defined below.  Defaults are initialized to ZERO.
*
**********************************************************************

        STRUCTURE       RDArgs,0
                STRUCT  RDA_Source,CS_SIZEOF    ; Select input source
                APTR    RDA_DAList              ; PRIVATE.
                LONG    RDA_Buffer              ; Optional string parsing space.
                LONG    RDA_BufSiz              ; Size of RDA_Buffer (0..n)
                APTR    RDA_ExtHelp             ; Optional extended help
                LONG    RDA_Flags               ; Flags for any required control
                LABEL   RDA_SIZEOF

        BITDEF  RDA,STDIN,0     ; Use "STDIN" rather than "COMMAND LINE"
        BITDEF  RDA,NOALLOC,1   ; If set, do not allocate extra string space.
        BITDEF  RDA,NOPROMPT,2  ; Disable reprompting for string input.

**********************************************************************
* Maximum number of template keywords which can be in a template passed
* to ReadArgs().  IMPLEMENTOR NOTE - must be a multiple of 4.
**********************************************************************
MAX_TEMPLATE_ITEMS      equ     100

**********************************************************************
* Maximum number of MULTIARG items returned by ReadArgs(), before
* an ERROR_LINE_TOO_LONG.  These two limitations are due to stack
* usage.  Applications should allow "a lot" of stack to use ReadArgs().
**********************************************************************
MAX_MULTIARGS   equ     128

        ENDC    ; DOS_RDARGS_I
```

dos/record.i

Page 1

```
        IFND DOS_RECORD_I
DOS_RECORD_I SET 1
**
**      $Filename: dos/record.i $
**      $Release: 2.04 $
**      $Revision: 36.5 $
**      $Date: 90/07/12 $
**
**      include file for record locking
**
**      (C) Copyright 1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**

     IFND  DOS_DOS_I
     INCLUDE "dos/dos.i"
     ENDC

* Modes for LockRecord/LockRecords()
REC_EXCLUSIVE           EQU     0
REC_EXCLUSIVE_IMMED     EQU     1
REC_SHARED              EQU     2
REC_SHARED_IMMED        EQU     3

* struct to be passed to LockRecords()/UnLockRecords()

 STRUCTURE RecordLock,0
        BPTR    rec_FH          ; filehandle
        ULONG   rec_Offset      ; offset in file
        ULONG   rec_Length      ; length of file to be locked
        ULONG   rec_Mode        ; Type of lock
 LABEL RecordLock_SIZEOF

        ENDC    ; DOS_RECORD_I

```

dos/var.i

Page 1

```
        IFND    DOS_VAR_I
DOS_VAR_I SET   1
**
**      $Filename: dos/var.i $
**      $Release: 2.04 $
**      $Revision: 36.11 $
**      $Date: 90/07/12 $
**
**      include file for dos local and environment variables
**
**      (C) Copyright 1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**

     IFND  EXEC_NODES_I
     INCLUDE "exec/nodes.i"
     ENDC

* the structure in the pr_LocalVars list
* Do NOT allocate yourself, use SetVar()!!! This structure may grow in
* future releases!  The list should be left in alphabetical order, and
* may have multiple entries with the same name but different types.

 STRUCTURE LocalVar,0
        STRUCT  lv_Node,LN_SIZE
        UWORD   lv_Flags
        APTR    lv_Value
        ULONG   lv_Len
 LABEL LocalVar_SIZEOF

*
* The lv_Flags bits are available to the application.  The unused
* lv_Node.ln_Pri bits are reserved for system use.
*

* bit definitions for lv_Node.ln_Type:

LV_VAR          EQU     0               ; a variable
LV_ALIAS        EQU     1               ; an alias
* to be or'ed into type:
LVB_IGNORE      EQU     7               ; ignore this entry on GetVar, etc
LVF_IGNORE      EQU     $80

* definitions of flags passed to GetVar()/SetVar()/DeleteVar()
* bit defs to be OR'ed with the type:
* item will be treated as a single line of text unless BINARY_VAR is used

        BITDEF  GV,GLOBAL_ONLY,8
        BITDEF  GV,LOCAL_ONLY,9
        BITDEF  GV,BINARY_VAR,10        ; treat as binary variable

        ENDC    ; DOS_VAR_I
        IFND    EXEC_ABLES_I
EXEC_ABLES_I    SET     1
**
**      $Filename: exec/ables.i $
**      $Release: 2.04 $
**      $Revision: 36.6 $
**      $Date: 90/05/10 $
**
**      Task switch and interrupt control macros
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

    IFND EXEC_TYPES_I
    INCLUDE "exec/types.i"
    ENDC        ; EXEC_TYPES_I

    IFND EXEC_EXECBASE_I
    INCLUDE "exec/execbase.i"
    ENDC        ; EXEC_EXECBASE_I


*----------------------------------------------------------------------
*
*   Interrupt Exclusion Macros.  Disable all tasks and interrupts.
*
*----------------------------------------------------------------------

INT_ABLES   MACRO               ; externals used by DISABLE and ENABLE
            XREF    _intena
            ENDM

;Disable interrupts.  Avoid use of DISABLE if at all possible.
;Please realize the danger of this macro!  Don't disable for long periods!
DISABLE     MACRO   ; [scratchReg],[NOFETCH] or have ExecBase in A6.
            IFC     '\1',''             ;Case 1: Assume A6=ExecBase
              MOVE.W  #$04000,_intena   ;(NOT IF_SETCLR)+IF_INTEN
              ADDQ.B  #1,IDNestCnt(A6)
              MEXIT
            ENDC
            IFC     '\2','NOFETCH'      ;Case 2: Assume \1=ExecBase
              MOVE.W  #$04000,_intena
              ADDQ.B  #1,IDNestCnt(\1)
              MEXIT
            ENDC
            IFNC    '\1',''             ;Case 3: Use \1 as scratch
              MOVE.L  4,\1              ;Get ExecBase
              MOVE.W  #$04000,_intena
              ADDQ.B  #1,IDNestCnt(\1)
              MEXIT
            ENDC
            ENDM

;Enable interrupts.  Please realize the danger of this macro!
ENABLE      MACRO   ; [scratchReg],[NOFETCH] or have ExecBase in A6.
            IFC     '\1',''             ;Case 1: Assume A6=ExecBase
              SUBQ.B  #1,IDNestCnt(A6)
              BGE.S   ENABLE\@
              MOVE.W  #$0C000,_intena   ;IF_SETCLR+IF_INTEN
ENABLE\@:
              MEXIT
            ENDC
            IFC     '\2','NOFETCH'      ;Case 2: Assume \1=ExecBase
              SUBQ.B  #1,IDNestCnt(\1)
              BGE.S   ENABLE\@
```

```
              MOVE.W  #$0C000,_intena
ENABLE\@:
              MEXIT
            ENDC
            IFNC    '\1',''             ;Case 3: Use \1 as scratch
              MOVE.L  4,\1              ;Get ExecBase
              SUBQ.B  #1,IDNestCnt(\1)
              BGE.S   ENABLE\@
              MOVE.W  #$0C000,_intena
ENABLE\@:
              MEXIT
            ENDC
            ENDM


*----------------------------------------------------------------------
*
*   Tasking Exclusion Macros.  Forbid all other tasks (but not interrupts)
*
*----------------------------------------------------------------------

TASK_ABLES  MACRO               ; externals used by FORBID and PERMIT
            XREF    _LVOPermit
            ENDM

;Prevent task switching (disables reschedule)
FORBID      MACRO   ; [scratchReg],[NOFETCH] or ExecBase in A6!
            IFC     '\1',''             ;Case 1: Assume A6=ExecBase
              ADDQ.B  #1,TDNestCnt(A6)
              MEXIT
            ENDC
            IFC     '\2','NOFETCH'      ;Case 2: Assume \1=ExecBase
              ADDQ.B  #1,TDNestCnt(\1)
              MEXIT
            ENDC
            IFNC    '\1',''             ;Case 3: Use \1 as scratch
              MOVE.L  4,\1      ;Get ExecBase
              ADDQ.B  #1,TDNestCnt(\1)
              MEXIT
            ENDC
            ENDM

;Enable task switching
PERMIT      MACRO   ; [saveFlag],[NOFETCH] or ExecBase in A6!
            IFC     '\1',''             ;Case 1: Assume A6=ExecBase
              JSR     _LVOPermit(A6)
              MEXIT
            ENDC
            IFC     '\2','NOFETCH'      ;Case 2: Assume \1=ExecBase
              EXG.L   A6,\1             ;put execbase in A6
              JSR     _LVOPermit(A6)    ;no registers touched.  A6=ExecBase
              EXG.L   A6,\1
              MEXIT
            ENDC
            IFNC    '\1',''             ;Case 2: save/restore A6
              MOVE.L  A6,-(SP)
              MOVE.L  4,A6
              JSR     _LVOPermit(A6)
              MOVE.L  (SP)+,A6
              MEXIT
            ENDC
            ENDM

        ENDC    ; EXEC_ABLES_I
```

```
    IFND EXEC_ALERTS_I
EXEC_ALERTS_I SET 1
**
**      $Filename: exec/alerts.i $
**      $Release: 2.04 $
**      $Revision: 36.17 $
**      $Date: 91/01/12 $
**
**      Alert numbers, as displayed by system crashes.
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

*********************************************************************
*
*  Format of the alert error number:
*
*    +---------------+----------------+--------------------------------+
*    |D|  SubSysId   |  General Error |    SubSystem Specific Error    |
*    +---------------+----------------+--------------------------------+
*     1    7 bits          8 bits                  16 bits
*
*                   D:  DeadEnd alert
*            SubSysId:  indicates ROM subsystem number.
*       General Error:  roughly indicates what the error was
*      Specific Error:  indicates more detail
*********************************************************************

*
*  Use this macro for causing an alert.  It is very sensitive to memory
*  corruption.... like stepping on location 4!  After the alert, it
*  will return.
*
*       A0/A1 and D0/D1 are destroyed
*
*
ALERT           MACRO   (alertNumber, [paramArray])
                movem.l d7/a5/a6,-(sp)
                move.l  #\1,d7
                IFNC    '\2',''
                  lea.l \2,a5
                ENDC
                move.l  4,a6
                jsr     _LVOAlert(a6)
                movem.l (sp)+,d7/a5/a6
                ENDM
*
*  Use this macro for dead end alerts that never return
*
DEADALERT       MACRO   (alertNumber, [paramArray])
                move.l  #\1,d7
                IFNC    '\2',''
                  lea.l \2,a5
                ENDC
                move.l  4,a6
                jsr     _LVOAlert(a6)   ; never returns
                ENDM


*********************************************************************
*
*  General Alerts
*
*  For example:  timer.device cannot open math.library:
*
```

```
*       ALERT  (AN_TimerDev!AG_OpenLib!AO_MathLib)       ;0x05038015
*
*********************************************************************

;------ alert types
AT_DeadEnd      equ $80000000
AT_Recovery     equ $00000000

;------ general purpose alert codes
AG_NoMemory     equ $00010000
AG_MakeLib      equ $00020000
AG_OpenLib      equ $00030000
AG_OpenDev      equ $00040000
AG_OpenRes      equ $00050000
AG_IOError      equ $00060000
AG_NoSignal     equ $00070000
AG_BadParm      equ $00080000
AG_CloseLib     equ $00090000   ;Usually too many closes
AG_CloseDev     equ $000A0000   ;or a mismatched close
AG_ProcCreate   equ $000B0000   ;Process creation failed

;------ alert objects:
AO_ExecLib      equ $00008001
AO_GraphicsLib  equ $00008002
AO_LayersLib    equ $00008003
AO_Intuition    equ $00008004
AO_MathLib      equ $00008005
AO_DOSLib       equ $00008007
AO_RAMLib       equ $00008008
AO_IconLib      equ $00008009
AO_ExpansionLib equ $0000800A
AO_DiskfontLib  equ $0000800B
AO_UtilityLib   equ $0000800C

AO_AudioDev     equ $00008010
AO_ConsoleDev   equ $00008011
AO_GamePortDev  equ $00008012
AO_KeyboardDev  equ $00008013
AO_TrackDiskDev equ $00008014
AO_TimerDev     equ $00008015

AO_CIARsrc      equ $00008020
AO_DiskRsrc     equ $00008021
AO_MiscRsrc     equ $00008022

AO_BootStrap    equ $00008030
AO_Workbench    equ $00008031
AO_DiskCopy     equ $00008032
AO_GadTools     equ $00008033
AO_Unknown      equ $00008035

*********************************************************************
*
*   Specific Alerts:
*
*   For example:   exec.library -- corrupted memory list
*
*           ALERT  AN_MemCorrupt        ;8100 0005
*
*********************************************************************

;------ exec.library
AN_ExecLib      equ $01000000
AN_ExcptVect    equ $01000001   ; 68000 exception vector checksum (obs.)
AN_BaseChkSum   equ $01000002   ; Execbase checksum bad (obs.)
AN_LibChkSum    equ $01000003   ; Library checksum failure
```

```

AN_MemCorrupt   equ $81000005   ; Corrupt memory list detected in FreeMem
AN_IntrMem      equ $81000006   ; No memory for interrupt servers
AN_InitAPtr     equ $01000007   ; InitStruct() of an APTR source (obs.)
AN_SemCorrupt   equ $01000008   ; A semaphore is in an illegal state
                                ; at RemSempahore()
AN_FreeTwice    equ $01000009   ; Freeing memory that is already free
AN_BogusExcpt   equ $8100000A   ; Illegal 68k exception taken (obs.)
AN_IOUsedTwice  equ $0100000B   ; Attempt to reuse active IORequest
AN_MemoryInsane equ $0100000C   ; Sanity check on memory list failed
                                ; during AvailMem(MEMF_LARGEST)
AN_IOAfterClose equ $0100000D   ; IO attempted on closed IORequest
AN_StackProbe   equ $0100000E   ; Stack appears to extend out of range
AN_BadFreeAddr  equ $0100000F   ; Memory header not located. [ Usually an
                                ; invalid address passed to FreeMem() ]

;------ graphics.library
AN_GraphicsLib  equ $02000000
AN_GfxNoMem     equ $82010000   ; graphics out of memory
AN_GfxNoMemMspc equ $82010001   ; MonitorSpec alloc, no memory
AN_LongFrame    equ $82010006   ; long frame, no memory
AN_ShortFrame   equ $82010007   ; short frame, no memory
AN_TextTmpRas   equ $02010009   ; text, no memory for TmpRas
AN_BltBitMap    equ $8201000A   ; BltBitMap, no memory
AN_RegionMemory equ $8201000B   ; regions, memory not available
AN_MakeVPort    equ $82010030   ; MakeVPort, no memory
AN_GfxNewError  equ $0200000C
AN_GfxFreeError equ $0200000D

AN_GfxNoLCM     equ $82011234   ; emergency memory not available

AN_ObsoleteFont equ $02000401   ; unsupported font description used

;------ layers.library
AN_LayersLib    equ $03000000
AN_LayersNoMem  equ $83010000   ; layers out of memory

;------ intuition.library
AN_Intuition    equ $04000000
AN_GadgetType   equ $84000001   ; unknown gadget type
AN_BadGadget    equ $04000001   ; Recovery form of AN_GadgetType
AN_CreatePort   equ $84010002   ; create port, no memory
AN_ItemAlloc    equ $04010003   ; item plane alloc, no memory
AN_SubAlloc     equ $04010004   ; sub alloc, no memory
AN_PlaneAlloc   equ $84010005   ; plane alloc, no memory
AN_ItemBoxTop   equ $84000006   ; item box top < RelZero
AN_OpenScreen   equ $84010007   ; open screen, no memory
AN_OpenScrnRast equ $84010008   ; open screen, raster alloc, no memory
AN_SysScrnType  equ $84000009   ; open sys screen, unknown type
AN_AddSWGadget  equ $8401000A   ; add SW gadgets, no memory
AN_OpenWindow   equ $8401000B   ; open window, no memory
AN_BadState     equ $8400000C   ; Bad State Return entering Intuition
AN_BadMessage   equ $8400000D   ; Bad Message received by IDCMP
AN_WeirdEcho    equ $8400000E   ; Weird echo causing incomprehension
AN_NoConsole    equ $8400000F   ; couldn't open the Console Device

;------ math.library
AN_MathLib      equ $05000000

;------ dos.library
AN_DOSLib       equ $07000000
AN_StartMem     equ $07010001   ; no memory at startup
AN_EndTask      equ $07000002   ; EndTask didn't
AN_QPktFail     equ $07000003   ; Qpkt failure
AN_AsyncPkt     equ $07000004   ; Unexpected packet received
AN_FreeVec      equ $07000005   ; Freevec failed
```

```
AN_DiskBlkSeq   equ $07000006   ; Disk block sequence error
AN_BitMap       equ $07000007   ; Bitmap corrupt
AN_KeyFree      equ $07000008   ; Key already free
AN_BadChkSum    equ $07000009   ; Invalid checksum
AN_DiskError    equ $0700000A   ; Disk Error
AN_KeyRange     equ $0700000B   ; Key out of range
AN_BadOverlay   equ $0700000C   ; Bad overlay
AN_BadInitFunc  equ $0700000D   ; Invalid init packet for cli/shell
AN_FileReclosed equ $0700000E   ; A filehandle was closed more than once

;------ ramlib.library
AN_RAMLib       equ $08000000
AN_BadSegList   equ $08000001   ; overlays are illegal for library segments

;------ icon.library
AN_IconLib      equ $09000000

;------ expansion.library
AN_ExpansionLib equ $0A000000
AN_BadExpansionFree     equ $0A000001   ;Freeed free region

;------ diskfont.library
AN_DiskfontLib  equ $0B000000

;------ audio.device
AN_AudioDev     equ $10000000

;------ console.device
AN_ConsoleDev   equ $11000000
AN_NoWindow     equ $11000001   ; Console can't open initial window

;------ gameport.device
AN_GamePortDev  equ $12000000

;------ keyboard.device
AN_KeyboardDev  equ $13000000

;------ trackdisk.device
AN_TrackDiskDev equ $14000000
AN_TDCalibSeek  equ $14000001   ; calibrate: seek error
AN_TDDelay      equ $14000002   ; delay: error on timer wait

;------ timer.device
AN_TimerDev     equ $15000000
AN_TMBadReq     equ $15000001   ; bad request
AN_TMBadSupply  equ $15000002   ; power supply -- no 50/60hz ticks

;------ cia.resource
AN_CIARsrc      equ $20000000

;------ disk.resource
AN_DiskRsrc     equ $21000000
AN_DRHasDisk    equ $21000001   ; get unit: already has disk
AN_DRIntNoAct   equ $21000002   ; interrupt: no active unit

;------ misc.resource
AN_MiscRsrc     equ $22000000

;------ bootstrap
AN_BootStrap    equ $30000000
AN_BootError    equ $30000001   ; boot code returned an error

;------ workbench
AN_Workbench                    equ $31000000
AN_NoFonts                      equ $B1000001
AN_WBBadStartupMsg1             equ $31000001
```

exec/alerts.i

```
AN_WBBadStartupMsg2             equ $31000002
AN_WBBadIOMsg                   equ $31000003

AN_WBInitPotionAllocDrawer      equ $B1010004
AN_WBCreateWBMenusCreateMenus1  equ $B1010005
AN_WBCreateWBMenusCreateMenus2  equ $B1010006
AN_WBLayoutWBMenusLayoutMenus   equ $B1010007
AN_WBAddToolMenuItem            equ $B1010008
AN_WBReLayoutToolMenu           equ $B1010009
AN_WBinitTimer                  equ $B101000A
AN_WBInitLayerDemon             equ $B101000B
AN_WBinitWbGels                 equ $B101000C
AN_WBInitScreenAndWindows1      equ $B101000D
AN_WBInitScreenAndWindows2      equ $B101000E
AN_WBInitScreenAndWindows3      equ $B101000F
AN_WBMAlloc                     equ $B1010010

;------ DiskCopy
AN_DiskCopy     equ $32000000

;------ toolkit for Intuition
AN_GadTools     equ $33000000

;------ System utility library
AN_UtilityLib   equ $34000000

;------ For use by any application that needs it
AN_Unknown      equ $35000000

    ENDC ;EXEC_ALERTS_I
```

exec/devices.i

```
        IFND    EXEC_DEVICES_I
EXEC_DEVICES_I  SET     1
**
**      $Filename: exec/devices.i $
**      $Release: 2.04 $
**      $Revision: 36.4 $
**      $Date: 90/05/10 $
**
**      Include file for use by Exec device drivers
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

    IFND EXEC_LIBRARIES_I
    INCLUDE "exec/libraries.i"
    ENDC        ; EXEC_LIBRARIES_I

    IFND EXEC_PORTS_I
    INCLUDE "exec/ports.i"
    ENDC        ; EXEC_PORTS_I


*----------------------------------------------------------------
*
*   Device Data Structure
*
*----------------------------------------------------------------

 STRUCTURE  DD,LIB_SIZE
    LABEL   DD_SIZE                     ; identical to library


*----------------------------------------------------------------
*
*   Suggested Unit Structure
*
*----------------------------------------------------------------

 STRUCTURE  UNIT,MP_SIZE                ; queue for requests
    UBYTE   UNIT_FLAGS
    UBYTE   UNIT_pad
    UWORD   UNIT_OPENCNT
    LABEL   UNIT_SIZE


*------ UNIT_FLAG definitions:

    BITDEF  UNIT,ACTIVE,0               ; driver is active
    BITDEF  UNIT,INTASK,1               ; running in driver's task

        ENDC    ; EXEC_DEVICES_I
```

exec/errors.i

```
        IFND    EXEC_ERRORS_I
EXEC_ERRORS_I   SET     1
**
**      $Filename: exec/errors.i $
**      $Release: 2.04 $
**      $Revision: 36.5 $
**      $Date: 90/05/27 $
**
**      Standard Device IO Errors (returned in io_Error)
**
**      (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**

IOERR_OPENFAIL          EQU -1  ; device/unit failed to open
IOERR_ABORTED           EQU -2  ; request terminated early [after AbortIO()]
IOERR_NOCMD             EQU -3  ; command not supported by device
IOERR_BADLENGTH         EQU -4  ; not a valid length (usually IO_LENGTH)
IOERR_BADADDRESS        EQU -5  ; invalid address (misaligned or bad range)
IOERR_UNITBUSY          EQU -6  ; device opens ok, but requested unit is busy
IOERR_SELFTEST          EQU -7  ; hardware failed self-test

ERR_OPENDEVICE          EQU  IOERR_OPENFAIL     ; Obsolete

        ENDC    ; EXEC_ERRORS_I
```

exec/exec.i

```
        IFND    EXEC_EXEC_I
EXEC_EXEC_I     SET     1
**
**      $Filename: exec/exec.i $
**      $Release: 2.04 $
**      $Revision: 36.7 $
**      $Date: 90/05/10 $
**
**      Include all other Exec include files in non-overlapping order.
**
**      (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**

        IFND    EXEC_TYPES_I
        INCLUDE "exec/types.i"
        ENDC
        INCLUDE "exec/macros.i"
        INCLUDE "exec/nodes.i"
        INCLUDE "exec/lists.i"
        INCLUDE "exec/alerts.i"
        INCLUDE "exec/errors.i"
        INCLUDE "exec/initializers.i"
        INCLUDE "exec/resident.i"
        INCLUDE "exec/strings.i"
        INCLUDE "exec/memory.i"
        INCLUDE "exec/tasks.i"
        INCLUDE "exec/ports.i"
        INCLUDE "exec/interrupts.i"
        INCLUDE "exec/semaphores.i"
        INCLUDE "exec/libraries.i"
        INCLUDE "exec/io.i"
        INCLUDE "exec/devices.i"
        INCLUDE "exec/execbase.i"
        INCLUDE "exec/ables.i"
;;;;;;;INCLUDE "exec/exec_lib.i"    ;special information

        ENDC    ; EXEC_EXEC_I
```

exec/exec_lib.i

Page 1

```
FUNCDEF Supervisor
FUNCDEF ExitIntr
FUNCDEF Schedule
FUNCDEF Reschedule
FUNCDEF Switch
FUNCDEF Dispatch
FUNCDEF Exception
FUNCDEF InitCode
FUNCDEF InitStruct
FUNCDEF MakeLibrary
FUNCDEF MakeFunctions
FUNCDEF FindResident
FUNCDEF InitResident
FUNCDEF Alert
FUNCDEF Debug
FUNCDEF Disable
FUNCDEF Enable
FUNCDEF Forbid
FUNCDEF Permit
FUNCDEF SetSR
FUNCDEF SuperState
FUNCDEF UserState
FUNCDEF SetIntVector
FUNCDEF AddIntServer
FUNCDEF RemIntServer
FUNCDEF Cause
FUNCDEF Allocate
FUNCDEF Deallocate
FUNCDEF AllocMem
FUNCDEF AllocAbs
FUNCDEF FreeMem
FUNCDEF AvailMem
FUNCDEF AllocEntry
FUNCDEF FreeEntry
FUNCDEF Insert
FUNCDEF AddHead
FUNCDEF AddTail
FUNCDEF Remove
FUNCDEF RemHead
FUNCDEF RemTail
FUNCDEF Enqueue
FUNCDEF FindName
FUNCDEF AddTask
FUNCDEF RemTask
FUNCDEF FindTask
FUNCDEF SetTaskPri
FUNCDEF SetSignal
FUNCDEF SetExcept
FUNCDEF Wait
FUNCDEF Signal
FUNCDEF AllocSignal
FUNCDEF FreeSignal
FUNCDEF AllocTrap
FUNCDEF FreeTrap
FUNCDEF AddPort
FUNCDEF RemPort
FUNCDEF PutMsg
FUNCDEF GetMsg
FUNCDEF ReplyMsg
FUNCDEF WaitPort
FUNCDEF FindPort
FUNCDEF AddLibrary
FUNCDEF RemLibrary
FUNCDEF OldOpenLibrary
FUNCDEF CloseLibrary
FUNCDEF SetFunction
```

exec/exec_lib.i

Page 2

```
FUNCDEF SumLibrary
FUNCDEF AddDevice
FUNCDEF RemDevice
FUNCDEF OpenDevice
FUNCDEF CloseDevice
FUNCDEF DoIO
FUNCDEF SendIO
FUNCDEF CheckIO
FUNCDEF WaitIO
FUNCDEF AbortIO
FUNCDEF AddResource
FUNCDEF RemResource
FUNCDEF OpenResource
FUNCDEF RawIOInit
FUNCDEF RawMayGetChar
FUNCDEF RawPutChar
FUNCDEF RawDoFmt
FUNCDEF GetCC
FUNCDEF TypeOfMem
FUNCDEF Procure
FUNCDEF Vacate
FUNCDEF OpenLibrary
FUNCDEF InitSemaphore
FUNCDEF ObtainSemaphore
FUNCDEF ReleaseSemaphore
FUNCDEF AttemptSemaphore
FUNCDEF ObtainSemaphoreList
FUNCDEF ReleaseSemaphoreList
FUNCDEF FindSemaphore
FUNCDEF AddSemaphore
FUNCDEF RemSemaphore
FUNCDEF SumKickData
FUNCDEF AddMemList
FUNCDEF CopyMem
FUNCDEF CopyMemQuick
FUNCDEF CacheClearU
FUNCDEF CacheClearE
FUNCDEF CacheControl
FUNCDEF CreateIORequest
FUNCDEF DeleteIORequest
FUNCDEF CreateMsgPort
FUNCDEF DeleteMsgPort
FUNCDEF ObtainSemaphoreShared
FUNCDEF AllocVec
FUNCDEF FreeVec
FUNCDEF CreatePrivatePool
FUNCDEF DeletePrivatePool
FUNCDEF AllocPooled
FUNCDEF FreePooled
FUNCDEF ExecReserved00
FUNCDEF ColdReboot
FUNCDEF StackSwap
FUNCDEF ChildFree
FUNCDEF ChildOrphan
FUNCDEF ChildStatus
FUNCDEF ChildWait
FUNCDEF ExecReserved01
FUNCDEF ExecReserved02
FUNCDEF ExecReserved03
FUNCDEF ExecReserved04
```

```
        IFND    EXEC_EXECBASE_I
EXEC_EXECBASE_I SET     1
**
**      $Filename: exec/execbase.i $
**      $Release: 2.04 $
**      $Revision: 36.12 $
**      $Date: 91/02/19 $
**
**      Definition of the exec.library base structure.
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

    IFND EXEC_TYPES_I
    INCLUDE "exec/types.i"
    ENDC        ; EXEC_TYPES_I

    IFND EXEC_LISTS_I
    INCLUDE "exec/lists.i"
    ENDC        ; EXEC_LISTS_I

    IFND EXEC_INTERRUPTS_I
    INCLUDE "exec/interrupts.i"
    ENDC        ; EXEC_INTERRUPTS_I

    IFND EXEC_LIBRARIES_I
    INCLUDE "exec/libraries.i"
    ENDC        ; EXEC_LIBRARIES_I


** Definition of the Exec library base structure (pointed to by location 4).
** Most fields are not to be viewed or modified by user programs.  Use
** extreme caution.
**
 STRUCTURE  ExecBase,LIB_SIZE          ; Standard library node

******* Static System Variables *********************************************

            UWORD       SoftVer ; kickstart release number (obs.)
            WORD        LowMemChkSum    ; checksum of 68000 trap vectors
            ULONG       ChkBase ; system base pointer complement
            APTR        ColdCapture     ; coldstart soft capture vector
            APTR        CoolCapture     ; coolstart soft capture vector
            APTR        WarmCapture     ; warmstart soft capture vector
            APTR        SysStkUpper     ; system stack base   (upper bound)
            APTR        SysStkLower     ; top of system stack (lower bound)
            ULONG       MaxLocMem       ; top of chip memory
            APTR        DebugEntry      ; global debugger entry point
            APTR        DebugData       ; global debugger data segment
            APTR        AlertData       ; alert data segment
            APTR        MaxExtMem       ; top of extended mem, or null if none

            WORD        ChkSum          ; for all of the above (minus 2)


******* Interrupt Related ***************************************************

            LABEL       IntVects
            STRUCT      IVTBE,IV_SIZE
            STRUCT      IVDSKBLK,IV_SIZE
            STRUCT      IVSOFTINT,IV_SIZE
            STRUCT      IVPORTS,IV_SIZE
            STRUCT      IVCOPER,IV_SIZE
            STRUCT      IVVERTB,IV_SIZE
```

```
            STRUCT      IVBLIT,IV_SIZE
            STRUCT      IVAUD0,IV_SIZE
            STRUCT      IVAUD1,IV_SIZE
            STRUCT      IVAUD2,IV_SIZE
            STRUCT      IVAUD3,IV_SIZE
            STRUCT      IVRBF,IV_SIZE
            STRUCT      IVDSKSYNC,IV_SIZE
            STRUCT      IVEXTER,IV_SIZE
            STRUCT      IVINTEN,IV_SIZE
            STRUCT      IVNMI,IV_SIZE


******* Dynamic System Variables *************************************

            APTR        ThisTask        ; pointer to current task (readable)

            ULONG       IdleCount       ; idle counter
            ULONG       DispCount       ; dispatch counter
            UWORD       Quantum ; time slice quantum
            UWORD       Elapsed ; current quantum ticks
            UWORD       SysFlags        ; misc internal system flags
            BYTE        IDNestCnt       ; interrupt disable nesting count
            BYTE        TDNestCnt       ; task disable nesting count

            UWORD       AttnFlags       ; special attention flags (readable)

            UWORD       AttnResched     ; rescheduling attention
            APTR        ResModules      ; pointer to resident module array
            APTR        TaskTrapCode    ; default task trap routine
            APTR        TaskExceptCode  ; default task exception code
            APTR        TaskExitCode    ; default task exit code
            ULONG       TaskSigAlloc    ; preallocated signal mask
            UWORD       TaskTrapAlloc   ; preallocated trap mask


******* System List Headers (private!) ********************************

            STRUCT      MemList,LH_SIZE
            STRUCT      ResourceList,LH_SIZE
            STRUCT      DeviceList,LH_SIZE
            STRUCT      IntrList,LH_SIZE
            STRUCT      LibList,LH_SIZE
            STRUCT      PortList,LH_SIZE
            STRUCT      TaskReady,LH_SIZE
            STRUCT      TaskWait,LH_SIZE

            STRUCT      SoftInts,SH_SIZE*5


******* Other Globals ******************************************************

            STRUCT      LastAlert,4*4

            ;------ these next two variables are provided to allow
            ;------ system developers to have a rough idea of the
            ;------ period of two externally controlled signals --
            ;------ the time between vertical blank interrupts and the
            ;------ external line rate (which is counted by CIA A's
            ;------ "time of day" clock).  In general these values
            ;------ will be 50 or 60, and may or may not track each
            ;------ other.  These values replace the obsolete AFB_PAL
            ;------ and AFB_50HZ flags.
            UBYTE       VBlankFrequency         ;(readable)
            UBYTE       PowerSupplyFrequency    ;(readable)

            STRUCT      SemaphoreList,LH_SIZE
```

exec/execbase.i

```

            ;------ these next two are to be able to kickstart into user ram.
            ;------ KickMemPtr holds a singly linked list of MemLists which
            ;------ will be removed from the memory list via AllocAbs.  If
            ;------ all the AllocAbs's succeeded, then the KickTagPtr will
            ;------ be added to the rom tag list.
            APTR        KickMemPtr     ; ptr to queue of mem lists
            APTR        KickTagPtr     ; ptr to rom tag queue
            APTR        KickCheckSum   ; checksum for mem and tags


******* V36 Exec additions start here ***************************************

            UWORD       ex_Pad0
            ULONG       ex_Reserved0
            APTR        ex_RamLibPrivate
            ;* The next ULONG contains the system "E" clock frequency,
            ;* expressed in Hertz.  The E clock is used as a timebase for
            ;* the Amiga's 8520 I/O chips. (E is connected to "02").
            ;* Typical values are 715909 for NTSC, or 709379 for PAL.
            ;*
            ULONG       ex_EClockFrequency ;(readable)
            ULONG       ex_CacheControl   ;Private to the CacheControl call
            ULONG       ex_TaskID         ;Next available task ID

            ULONG       ex_PuddleSize
            ULONG       ex_PoolThreshold
            STRUCT      ex_PublicPool,MLN_SIZE

            APTR        ex_MMULock        ;Private

            STRUCT      ex_Reserved,12

            LABEL       SYSBASESIZE


******* Bit defines for AttnFlags (see above) *******************************

*  Processors and Co-processors:
        BITDEF  AF,68010,0      ; also set for 68020
        BITDEF  AF,68020,1      ; also set for 68030
        BITDEF  AF,68030,2      ; also set for 68040
        BITDEF  AF,68040,3
        BITDEF  AF,68881,4      ; also set for 68882
        BITDEF  AF,68882,5
        BITDEF  AF,68851_MMU,6  ; 68851 MMU present
;       BITDEF  AF,RESERVED8,8
;       BITDEF  AF,RESERVED9,9


******* Selected bit definitions for Cache manipulation calls **************

        BITDEF  CACR,EnableI,0         ;Enable instruction cache
        BITDEF  CACR,FreezeI,1         ;Freeze instruction cache
        BITDEF  CACR,ClearI,3          ;Clear instruction cache
        BITDEF  CACR,IBE,4             ;Instruction burst enable
        BITDEF  CACR,EnableD,8         ;68030 Enable data cache
        BITDEF  CACR,FreezeD,9         ;68030 Freeze data cache
        BITDEF  CACR,ClearD,11         ;68030 Clear data cache
        BITDEF  CACR,DBE,12            ;68030 Data burst enable
        BITDEF  CACR,WriteAllocate,13  ;68030 Write-Allocate mode (must
                                       ;always be set)
        BITDEF  CACR,CopyBack,31       ;Master enable for copyback caches

        ENDC    ; EXEC_EXECBASE_I
```

exec/initializers.i

```
        IFND    EXEC_INITIALIZERS_I
EXEC_INITIALIZERS_I     SET     1
**
**      $Filename: exec/initializers.i $
**      $Release: 2.04 $
**      $Revision: 36.5 $
**      $Date: 90/05/10 $
**
**      Macros for creating InitStruct() tables
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

INITBYTE        MACRO   ; &offset,&value
                IFLE    (\1)-255        ;If offset <=255
                DC.B    $a0,\1          ;use byte offset
                DC.B    \2,0
                MEXIT                   ;exit early
                ENDC
                DC.B    $e0,0
                DC.W    \1
                DC.B    \2,0
                ENDM

INITWORD        MACRO   ; &offset,&value
                IFLE    (\1)-255        ;If offset <=255
                DC.B    $90,\1          ;use byte offset
                DC.W    \2
                MEXIT                   ;exit early
                ENDC
                DC.B    $d0,0
                DC.W    \1
                DC.W    \2
                ENDM

INITLONG        MACRO   ; &offset,&value
                IFLE    (\1)-255        ;If offset <=255
                DC.B    $80,\1          ;use byte offset
                DC.L    \2
                MEXIT                   ;exit early
                ENDC
                DC.B    $c0,0
                DC.W    \1
                DC.L    \2
                ENDM

;size=source size 0=long, 1=word, 2=byte, 3=illegal.
;offset=offset from memory base to put data
;value=unused
;count=number of source items to copy, minus one
;follow this macro with the proper sized data (dc.b,dc.w,dc.l,etc.)
INITSTRUCT      MACRO   ; &size,&offset,&value,&count
                DS.W    0
                IFC     '\4',''
COUNT\@ SET     0
                ENDC
                IFNC    '\4',''
COUNT\@ SET     \4
                ENDC
CMD\@           SET     (((\1)<<4)!COUNT\@)
                IFLE    (\2)-255        ;byte offset large enough?
                DC.B    (CMD\@)!$80
                DC.B    \2
                MEXIT
                ENDC
```

exec/initializers.i

```
                DC.B    CMD\@!$0C0      ;byte too small, use 24-bit offset.
                DC.B    (((\2)>>16)&$0FF)
                DC.W    ((\2)&$0FFFF)
                ENDM

        ENDC    ; EXEC_INITIALIZERS_I
```

exec/interrupts.i

```
        IFND    EXEC_INTERRUPTS_I
EXEC_INTERRUPTS_I       SET     1
**
**      $Filename: exec/interrupts.i $
**      $Release: 2.04 $
**      $Revision: 36.5 $
**      $Date: 90/05/10 $
**
**      Callback structures used by hardware & software interrupts
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved

    IFND EXEC_NODES_I
    INCLUDE "exec/nodes.i"
    ENDC        ; EXEC_NODES_I

    IFND EXEC_LISTS_I
    INCLUDE "exec/lists.i"
    ENDC        ; EXEC_LISTS_I

*---------------------------------------------------------------
*
*   Interrupt Structure
*
*---------------------------------------------------------------

 STRUCTURE  IS,LN_SIZE
    APTR    IS_DATA
    APTR    IS_CODE
    LABEL   IS_SIZE

*--------------------------------------------------------------------
*
*   Exec Internal Interrupt Vectors
*
*--------------------------------------------------------------------

 STRUCTURE  IV,0
    APTR    IV_DATA
    APTR    IV_CODE
    APTR    IV_NODE
    LABEL   IV_SIZE

*------ System Flag bits (in SysBase.SysFlags )

    BITDEF  S,SAR,15        ; scheduling attention required (TOP BIT)
    BITDEF  S,TQE,14        ; time quantum expended -- time to resched
    BITDEF  S,SINT,13

*--------------------------------------------------------------------
*
*   Software Interrupt List Headers
*
*--------------------------------------------------------------------
 STRUCTURE  SH,LH_SIZE
    UWORD   SH_PAD
    LABEL   SH_SIZE

SIH_PRIMASK EQU        $0F0
SIH_QUEUES  EQU        5

** this is a fake INT definition, used only for AddIntServer and the like
        BITDEF  INT,NMI,15

        ENDC    ; EXEC_INTERRUPTS_I
        IFND    EXEC_IO_I
EXEC_IO_I       SET     1
**
**      $Filename: exec/io.i $
**      $Release: 2.04 $
**      $Revision: 36.4 $
**      $Date: 90/05/10 $
**
**      Message structures used for device communication
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

    IFND EXEC_PORTS_I
    INCLUDE "exec/ports.i"
    ENDC        ; EXEC_PORTS_I

    IFND EXEC_LIBRARIES_I
    INCLUDE "exec/libraries.i"
    ENDC        ; EXEC_LIBRARIES_I


;----------------------------------------------------------------
;
;   IO Request Structures
;
;----------------------------------------------------------------

;------ Required portion of IO request:

 STRUCTURE  IO,MN_SIZE
    APTR    IO_DEVICE                   ; device node pointer
    APTR    IO_UNIT                     ; unit (driver private)
    UWORD   IO_COMMAND                  ; device command
    UBYTE   IO_FLAGS                    ; special flags
    BYTE    IO_ERROR                    ; error or warning code
    LABEL   IO_SIZE


;------ Standard IO request extension:

    ULONG   IO_ACTUAL                   ; actual # of bytes transfered
    ULONG   IO_LENGTH                   ; requested # of bytes transfered
    APTR    IO_DATA                     ; pointer to data area
    ULONG   IO_OFFSET                   ; offset for seeking devices
    LABEL   IOSTD_SIZE


;------ IO_FLAGS bit definitions:

    BITDEF  IO,QUICK,0                  ; complete IO quickly


;----------------------------------------------------------------
;
;   Standard Device Library Functions
;
;----------------------------------------------------------------

            LIBINIT

            LIBDEF  DEV_BEGINIO ; process IO request
            LIBDEF  DEV_ABORTIO ; abort IO request
```

```
;----------------------------------------------------------------
;
;   IO Function Macros
;
;----------------------------------------------------------------

BEGINIO     MACRO
            LINKLIB DEV_BEGINIO,IO_DEVICE(A1)
            ENDM

ABORTIO     MACRO
            LINKLIB DEV_ABORTIO,IO_DEVICE(A1)
            ENDM


;----------------------------------------------------------------
;
;   Standard Device Command Definitions
;
;----------------------------------------------------------------

;------ Command definition macro:
DEVINIT     MACRO   ; [baseOffset]
            IFC     '\1',''
CMD_COUNT   SET     CMD_NONSTD
            ENDC
            IFNC    '\1',''
CMD_COUNT   SET     \1
            ENDC
            ENDM

DEVCMD      MACRO   ; cmdname
\1          EQU     CMD_COUNT
CMD_COUNT   SET     CMD_COUNT+1
            ENDM


;------ Standard device commands:

            DEVINIT 0

            DEVCMD  CMD_INVALID ; invalid command
            DEVCMD  CMD_RESET               ; reset as if just inited
            DEVCMD  CMD_READ                ; standard read
            DEVCMD  CMD_WRITE               ; standard write
            DEVCMD  CMD_UPDATE              ; write out all buffers
            DEVCMD  CMD_CLEAR               ; clear all buffers
            DEVCMD  CMD_STOP                ; hold current and queued
            DEVCMD  CMD_START               ; restart after stop
            DEVCMD  CMD_FLUSH               ; abort entire queue


;------ First non-standard device command value:

            DEVCMD  CMD_NONSTD

        ENDC    ; EXEC_IO_I
```

exec/libraries.i

```
        IFND    EXEC_LIBRARIES_I
EXEC_LIBRARIES_I        SET     1
**
**      $Filename: exec/libraries.i $
**      $Release: 2.04 $
**      $Revision: 36.11 $
**      $Date: 91/02/11 $
**
**      Definitions for use when creating or using Exec libraries
**
**      (C) Copyright 1985,1986,1987,1988,1989,1991 Commodore-Amiga, Inc.
**          All Rights Reserved
**

    IFND EXEC_NODES_I
    INCLUDE "exec/nodes.i"
    ENDC        ; EXEC_NODES_I


*------ Special Constants ---------------------------------------
LIB_VECTSIZE    EQU     6               ;Each library entry takes 6 bytes
LIB_RESERVED    EQU     4               ;Exec reserves the first 4 vectors
LIB_BASE        EQU     -LIB_VECTSIZE
LIB_USERDEF     EQU     LIB_BASE-(LIB_RESERVED*LIB_VECTSIZE) ;First user func
LIB_NONSTD      EQU     LIB_USERDEF


*----------------------------------------------------------------
*
*   Library Definition Macros (for creating libraries)
*
*----------------------------------------------------------------

*------ LIBINIT initializes the base offset for using the "LIBDEF" macro:
LIBINIT     MACRO   ; [baseOffset]
            IFC     '\1',''
COUNT_LIB   SET     LIB_USERDEF
            ENDC
            IFNC    '\1',''
COUNT_LIB   SET     \1
            ENDC
            ENDM

*------ LIBDEF is used to define each library function entry:
LIBDEF      MACRO   ;libraryFunctionSymbol
\1          EQU     COUNT_LIB
COUNT_LIB   SET     COUNT_LIB-LIB_VECTSIZE
            ENDM

*------ FUNCDEF is used to parse library offset tables.  Many applications
*------ need a special version of FUNCDEF - you provide your own macro
*------ to match your needs.  Here is an example:
*
*       FUNCDEF          MACRO
*       _LVO\1           EQU    FUNC_CNT
*       FUNC_CNT         SET    FUNC_CNT-6      * Standard offset-6 bytes each
*       FUNC_CNT         EQU    LIB_USERDEF     * Skip 4 standard vectors
*                        ENDM

*----------------------------------------------------------------
*
*   Standard Library Functions
*
*----------------------------------------------------------------

    LIBINIT LIB_BASE
```

exec/libraries.i

```

    LIBDEF  LIB_OPEN
    LIBDEF  LIB_CLOSE
    LIBDEF  LIB_EXPUNGE ; must exist in all libraries
    LIBDEF  LIB_EXTFUNC ; for future expansion - must return zero.


*----------------------------------------------------------------
*
*   Library Base Structure Definition
*   Also used for Devices and some Resources
*
*----------------------------------------------------------------

 STRUCTURE LIB,LN_SIZE
    UBYTE   LIB_FLAGS                    ; see below
    UBYTE   LIB_pad                      ; must be zero
    UWORD   LIB_NEGSIZE          ; number of bytes before LIB
    UWORD   LIB_POSSIZE          ; number of bytes after LIB
    UWORD   LIB_VERSION          ; major
    UWORD   LIB_REVISION                 ; minor
    APTR    LIB_IDSTRING                 ; ASCII identification
    ULONG   LIB_SUM                      ; the system-calculated checksum
    UWORD   LIB_OPENCNT          ; number of current opens
    LABEL   LIB_SIZE    ;Warning: Size is not a longword multiple!

*------ LIB_FLAGS bit definitions (all others are system reserved)
    BITDEF  LIB,SUMMING,0  ; system is currently checksumming
    BITDEF  LIB,CHANGED,1  ; something has changed the library since last sum
    BITDEF  LIB,SUMUSED,2  ; indicates if the library allows checksumming
    BITDEF  LIB,DELEXP,3   ; delayed expunge flag (for use by library)
    BITDEF  LIB,EXPOCNT,4  ; special system expunge flag.


*----------------------------------------------------------------------
*
*       Function Invocation Macros (for calling existing, opened, libraries)
*       Also see exec/macros.i
*
*----------------------------------------------------------------------

*------ CALLLIB for calling functions where A6 is already correct:

CALLLIB     MACRO   ; functionOffset
        IFGT NARG-1
            FAIL    !!! CALLLIB MACRO - too many arguments !!!
        ENDC
            JSR     \1(A6)
            ENDM


*------ LINKLIB for calling functions where A6 is incorrect:

LINKLIB     MACRO   ; functionOffset,libraryBase
        IFGT NARG-2
            FAIL    !!! LINKLIB MACRO - too many arguments !!!
        ENDC
            MOVE.L  A6,-(SP)
            MOVE.L  \2,A6
            JSR     \1(A6)
            MOVE.L  (SP)+,A6
            ENDM

        ENDC    ; EXEC_LIBRARIES_I
        IFND    EXEC_LISTS_I
EXEC_LISTS_I    SET     1
**
**      $Filename: exec/lists.i $
**      $Release: 2.04 $
**      $Revision: 36.10 $
**      $Date: 91/02/19 $
**
**      Definitions and macros for use with Exec lists.  Most of the
**      macros require ownership or locking of the list before use.
**
**      (C) Copyright 1985,1986,1987,1988,1989,1991 Commodore-Amiga, Inc.
**          All Rights Reserved
**

        IFND EXEC_NODES_I
        INCLUDE "exec/nodes.i"
        ENDC    ; EXEC_NODES_I


*---------------------------------------------------------------------

*
* Full featured list header
*
   STRUCTURE    LH,0
        APTR    LH_HEAD
        APTR    LH_TAIL
        APTR    LH_TAILPRED
        UBYTE   LH_TYPE
        UBYTE   LH_pad
        LABEL   LH_SIZE ;word aligned

*
* Minimal List Header - no type checking (best for most applications)
*
   STRUCTURE    MLH,0
        APTR    MLH_HEAD
        APTR    MLH_TAIL
        APTR    MLH_TAILPRED
        LABEL   MLH_SIZE ;longword aligned

*---------------------------------------------------------------------

;Prepare a list header for use
NEWLIST     MACRO   ; list
            MOVE.L  \1,LH_TAILPRED(\1)
            ADDQ.L  #4,\1           ;Get address of LH_TAIL
            CLR.L   (\1)            ;Clear LH_TAIL
            MOVE.L  \1,-(\1)        ;Address of LH_TAIL to LH_HEAD
            ENDM

;Test if list is empty (list address in register)
;This operation is safe at any time - no list arbitration needed.
TSTLIST     MACRO   ; [list]
            IFGT    NARG-1
               FAIL    !!! TSTLIST - Too many arguments !!!
            ENDC
            IFC     '\1',''
            CMP.L   LH_TAIL+LN_PRED(A0),A0
            ENDC
            IFNC    '\1',''
            CMP.L   LH_TAIL+LN_PRED(\1),\1
            ENDC
            ENDM

```

```
;Test if list is empty (from effective address of list)
;list arbitration required.
TSTLST2     MACRO   ;EA of list,=node
            MOVE.L  \1,\2
            TST.L   (\2)
            ENDM

;Get next in list
SUCC        MACRO   ; node,=succ
            MOVE.L  (\1),\2
            ENDM

;Get previous in list
PRED        MACRO   ; node,=pred
            MOVE.L  LN_PRED(\1),\2
            ENDM

;If empty, branch
IFEMPTY     MACRO   ; list,label
            CMP.L   LH_TAIL+LN_PRED(\1),\1
            BEQ     \2
            ENDM

;If not empty, branch
IFNOTEMPTY  MACRO   ; list,label
            CMP.L   LH_TAIL+LN_PRED(\1),\1
            BNE     \2
            ENDM

;Get next node, test if at end
TSTNODE     MACRO   ; node,=next
            MOVE.L  (\1),\2
            TST.L   (\2)
            ENDM

;Get next, go to exit label if at end
NEXTNODE    MACRO   ; next=next,=current,exit_label ([.s],DX,AX,DISP16)
            MOVE.L  \1,\2
            MOVE.L  (\2),\1
            IFC     '\0',''         ;Check extension
            BEQ     \3
            ENDC
            IFNC    '\0',''
            BEQ.S   \3
            ENDC
            ENDM

;Add to head of list
ADDHEAD     MACRO   ; A0-list(destroyed) A1-node D0-(destroyed)
            MOVE.L  (A0),D0
            MOVE.L  A1,(A0)
            MOVEM.L D0/A0,(A1)
            MOVE.L  D0,A0
            MOVE.L  A1,LN_PRED(A0)
            ENDM

;Add to tail of list
ADDTAIL     MACRO   ; A0-list(destroyed) A1-node D0-(destroyed)
            ADDQ.L  #LH_TAIL,A0
            MOVE.L  LN_PRED(A0),D0
            MOVE.L  A1,LN_PRED(A0)
            MOVE.L  A0,(A1)
            MOVE.L  D0,LN_PRED(A1)
            MOVE.L  D0,A0
            MOVE.L  A1,(A0)
            ENDM
```

exec/lists.i

Page 3

```
;Remove node from whatever list it is in
REMOVE      MACRO   ; A0-(destroyed)  A1-node(destroyed)
            MOVE.L  (A1),A0
            MOVE.L  LN_PRED(A1),A1
            MOVE.L  A0,(A1)
            MOVE.L  A1,LN_PRED(A0)
            ENDM

;Remove node from head of list
REMHEAD     MACRO   ; A0-list A1-(destroyed) D0=node
            MOVE.L  (A0),A1
            MOVE.L  (A1),D0
            BEQ.S   REMHEAD\@
            MOVE.L  D0,(A0)
            EXG.L   D0,A1
            MOVE.L  A0,LN_PRED(A1)
REMHEAD\@
            ENDM

;Remove head quickly
;       Useful when a scratch register is available, and
;       list is known to contain at least one node.
REMHEADQ    MACRO   ; list,=node,scratchReg-(destroyed)
            MOVE.L  (\1),\2
            MOVE.L  (\2),\3
            MOVE.L  \3,(\1)
            MOVE.L  \1,LN_PRED(\3)
            ENDM

;Remove node from tail of list
REMTAIL     MACRO   ; A0-list A1-(destroyed) D0=node
            MOVE.L  LH_TAIL+LN_PRED(A0),A1
            MOVE.L  LN_PRED(A1),D0
            BEQ.S   REMTAIL\@
            MOVE.L  D0,LH_TAIL+LN_PRED(A0)
            EXG.L   D0,A1
            MOVE.L  A0,(A1)
            ADDQ.L  #4,(A1)
REMTAIL\@
            ENDM

        ENDC    ; EXEC_LISTS_I
```

exec/macros.i

Page 1

```
        IFND    EXEC_MACROS_I
EXEC_MACROS_I   SET     1
**
**      $Filename: exec/macros.i $
**      $Release: 2.04 $
**      $Revision: 36.8 $
**      $Date: 90/11/01 $
**
**      Handy macros for assembly language programmers.
**
**      (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**

                IFND    DEBUG_DETAIL
DEBUG_DETAIL    SET     0       ;Detail level of debugging.  Zero for none.
                ENDC


JSRLIB          MACRO   ;FunctionName
                XREF    _LVO\1
                jsr     _LVO\1(a6)
                ENDM

JMPLIB          MACRO   ;FunctionName
                XREF    _LVO\1
                jmp     _LVO\1(a6)
                ENDM

BSRSELF MACRO
                XREF    \1
                bsr     \1
                ENDM

BRASELF MACRO
                XREF    \1
                bra     \1
                ENDM

BLINK           MACRO
                IFNE    DEBUG_DETAIL
                  bchg.b #1,$bfe001  ;Toggle the power LED
                ENDC
                ENDM

TRIGGER         MACRO   ;<level> Trigger a hardware state analyzer
                IFGE    DEBUG_DETAIL-\1
                  move.w #$5555,$2fe
                ENDC
                ENDM

CLEAR           MACRO
                moveq.l #0,\1
                ENDM

CLEARA          MACRO
                suba.l  \1,\1   ;Quick way to put zero in an address register
                ENDM

************************************************************************
                IFND    PRINTF
PRINTF          MACRO   ; level,<string>,...
                IFGE    DEBUG_DETAIL-\1
                XREF    kprint_macro
PUSHCOUNT       SET     0

```

exec/macros.i Page 2

```
                IFNC    '\9',''
                move.l  \9,-(sp)
PUSHCOUNT       SET     PUSHCOUNT+4
                ENDC

                IFNC    '\8',''
                move.l  \8,-(sp)
PUSHCOUNT       SET     PUSHCOUNT+4
                ENDC

                IFNC    '\7',''
                move.l  \7,-(sp)
PUSHCOUNT       SET     PUSHCOUNT+4
                ENDC

                IFNC    '\6',''
                move.l  \6,-(sp)
PUSHCOUNT       SET     PUSHCOUNT+4
                ENDC

                IFNC    '\5',''
                move.l  \5,-(sp)
PUSHCOUNT       SET     PUSHCOUNT+4
                ENDC

                IFNC    '\4',''
                move.l  \4,-(sp)
PUSHCOUNT       SET     PUSHCOUNT+4
                ENDC

                IFNC    '\3',''
                move.l  \3,-(sp)
PUSHCOUNT       SET     PUSHCOUNT+4
                ENDC

                movem.l a0/a1,-(sp)
                lea.l   PSS\@(pc),A0
                lea.l   4*2(SP),A1
                BSR     kprint_macro
                movem.l (sp)+,a0/a1
                bra.s   PSE\@

PSS\@           dc.b    \2
                IFEQ    (\1&1)  ;If even, add CR/LF par...
                   dc.b 13,10
                ENDC
                dc.b    0
                ds.w    0
PSE\@
                lea.l   PUSHCOUNT(sp),sp
                ENDC    ;IFGE   DEBUG_DETAIL-\1
                ENDM    ;PRINTF MACRO
                ENDC    ;IFND   PRINTF


;------------------------------------------------------------------------
;Push a set of registers onto the stack - undo with POPM.  This prevents
;the need to update or synchronize two MOVEM instructions.  These macros
;assume an optimizing assembler that will convert single register  MOVEM
;to MOVE.  Because the REG assignment can't be reset, these macros do
;not nest.
;
;           PUSHM   d2/a2/a5
;           ...code...
;           POPM
;           RTS
```

exec/macros.i Page 3

```
;
PUSHM_COUNT                     SET     0
PUSHM                           MACRO
                                IFGT    NARG-1
                                FAIL    !!!! TOO MANY ARGUMENTS TO PUSHM !!!!
                                ENDC
PUSHM_COUNT                     SET     PUSHM_COUNT+1
PUSHM_\*VALOF(PUSHM_COUNT)      REG     \1
                                movem.l PUSHM_\*VALOF(PUSHM_COUNT),-(sp)
                                ENDM

;
;Undo most recent PUSHM.  'POPM NOBUMP' allows multiple exit points.
;
POPM                            MACRO
                                movem.l (sp)+,PUSHM_\*VALOF(PUSHM_COUNT)
                                IFNC    '\1','NOBUMP'
PUSHM_COUNT                     SET     PUSHM_COUNT+1   ;error if re-used
                                ENDC
                                ENDM
;----------------------------------------------------------------------------


        ENDC    ; EXEC_MACROS_I
        IFND    EXEC_MEMORY_I
EXEC_MEMORY_I   SET     1
**
**      $Filename: exec/memory.i $
**      $Release: 2.04 $
**      $Revision: 36.10 $
**      $Date: 90/06/11 $
**
**      Definitions and structures used by the memory allocation system
**
**      (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**

    IFND EXEC_NODES_I
    INCLUDE "exec/nodes.i"
    ENDC        ; EXEC_NODES_I


*----------------------------------------------------------------------
*
*   Memory List Structures
*
*----------------------------------------------------------------------
*
*   A memory list appears in two forms:  One is a requirements list
*   the other is a list of already allocated memory.  The format is
*   the same, with the reqirements/address field occupying the same
*   position.
*
*   The format is a linked list of ML structures each of which has
*   an array of ME entries.
*
*----------------------------------------------------------------------

 STRUCTURE ML,LN_SIZE
    UWORD   ML_NUMENTRIES           ; The number of ME structures that follow
    LABEL   ML_ME                   ; where the ME structures begin
    LABEL   ML_SIZE     ;Note: does NOT include any "ME" structures.


 STRUCTURE ME,0
    LABEL   ME_REQS                 ; the AllocMem requirements
    APTR    ME_ADDR                 ; the address of this block (an alias
                                    ;   for the same location as ME_REQS)
    ULONG   ME_LENGTH               ; the length of this region
    LABEL   ME_SIZE


*------ memory options:
*------ see the AllocMem() documentation for details------*

MEMF_ANY        EQU 0           ;Any type of memory will do
    BITDEF  MEM,PUBLIC,0
    BITDEF  MEM,CHIP,1
    BITDEF  MEM,FAST,2
    BITDEF  MEM,LOCAL,8
    BITDEF  MEM,24BITDMA,9      ;DMAable memory within 24 bits of address

    BITDEF  MEM,CLEAR,16
    BITDEF  MEM,LARGEST,17
    BITDEF  MEM,REVERSE,18
    BITDEF  MEM,TOTAL,19        ;AvailMem: return total size of memory


*------ alignment rules for a memory block:
```

```

MEM_BLOCKSIZE   EQU 8
MEM_BLOCKMASK   EQU (MEM_BLOCKSIZE-1)


*----------------------------------------------------------------------
*
*   Memory Region Header
*
*----------------------------------------------------------------------

 STRUCTURE  MH,LN_SIZE                  ; (LN_TYPE will be set to NT_MEMORY)
    UWORD   MH_ATTRIBUTES               ; characteristics of this region
    APTR    MH_FIRST                    ; first free region
    APTR    MH_LOWER                    ; lower memory bound
    APTR    MH_UPPER                    ; upper memory bound+1
    ULONG   MH_FREE                     ; number of free bytes
    LABEL   MH_SIZE


*----------------------------------------------------------------------
*
*   Memory Chunk
*
*----------------------------------------------------------------------

 STRUCTURE  MC,0
    APTR    MC_NEXT                     ; ptr to next chunk
    ULONG   MC_BYTES                    ; chunk byte size
    APTR    MC_SIZE

        ENDC    ; EXEC_MEMORY_I
```

exec/nodes.i

```
        IFND EXEC_NODES_I
EXEC_NODES_I    SET 1
**
**      $Filename: exec/nodes.i $
**      $Release: 2.04 $
**      $Revision: 36.10 $
**      $Date: 91/01/09 $
**
**      Nodes & Node type identifiers.
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved

    IFND EXEC_TYPES_I
    INCLUDE "exec/types.i"
    ENDC ; EXEC_TYPES_I

*
*   List Node Structure.  Each member in a list starts with a Node
*
   STRUCTURE    LN,0    ; List Node
        APTR    LN_SUCC ; Pointer to next (successor)
        APTR    LN_PRED ; Pointer to previous (predecessor)
        UBYTE   LN_TYPE
        BYTE    LN_PRI  ; Priority, for sorting
        APTR    LN_NAME ; ID string, null terminated
        LABEL   LN_SIZE ; Note: word aligned

; minimal node -- no type checking possible
   STRUCTURE    MLN,0   ; Minimal List Node
        APTR    MLN_SUCC
        APTR    MLN_PRED
        LABEL   MLN_SIZE

**
** Note: Newly initialized IORequests, and software interrupt structures
** used with Cause(), should have type NT_UNKNOWN.  The OS will assign a type
** when they are first used.
**
;------ Node Types for LN_TYPE

NT_UNKNOWN      EQU 0
NT_TASK EQU 1   ; Exec task
NT_INTERRUPT    EQU 2
NT_DEVICE       EQU 3
NT_MSGPORT      EQU 4
NT_MESSAGE      EQU 5   ; Indicates message currently pending
NT_FREEMSG      EQU 6
NT_REPLYMSG     EQU 7   ; Message has been replied
NT_RESOURCE     EQU 8
NT_LIBRARY      EQU 9
NT_MEMORY       EQU 10
NT_SOFTINT      EQU 11  ; Internal flag used by SoftInts
NT_FONT EQU 12
NT_PROCESS      EQU 13  ; AmigaDOS Process
NT_SEMAPHORE    EQU 14
NT_SIGNALSEM    EQU 15  ; signal semaphores
NT_BOOTNODE     EQU 16
NT_KICKMEM      EQU 17
NT_GRAPHICS     EQU 18
NT_DEATHMESSAGE EQU 19

NT_USER         EQU 254 ; User node types work down from here
NT_EXTENDED     EQU 255

        ENDC    ;EXEC_NODES_I
```

exec/ports.i

```
        IFND    EXEC_PORTS_I
EXEC_PORTS_I    SET     1
**
**      $Filename: exec/ports.i $
**      $Release: 2.04 $
**      $Revision: 36.5 $
**      $Date: 90/05/10 $
**
**      Message ports and Messages.
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

    IFND EXEC_NODES_I
    INCLUDE "exec/nodes.i"
    ENDC        ; EXEC_NODES_I

    IFND EXEC_LISTS_I
    INCLUDE "exec/lists.i"
    ENDC        ; EXEC_LISTS_I


*----------------------------------------------------------------
*
*   Message Port Structure
*
*----------------------------------------------------------------

 STRUCTURE  MP,LN_SIZE
    UBYTE   MP_FLAGS
    UBYTE   MP_SIGBIT                   ; signal bit number
    APTR    MP_SIGTASK                  ; object to be signalled
    STRUCT  MP_MSGLIST,LH_SIZE          ; message linked list
    LABEL   MP_SIZE


*------ unions:
MP_SOFTINT      EQU  MP_SIGTASK


*------ MP_FLAGS: Port arrival actions (PutMsg)
PF_ACTION       EQU  3  ; Mask
PA_SIGNAL       EQU  0  ; Signal task in MP_SIGTASK
PA_SOFTINT      EQU  1  ; Signal SoftInt in MP_SOFTINT/MP_SIGTASK
PA_IGNORE       EQU  2  ; Ignore arrival


*----------------------------------------------------------------
*
*   Message Structure
*
*----------------------------------------------------------------

 STRUCTURE  MN,LN_SIZE
    APTR    MN_REPLYPORT            ; message reply port
    UWORD   MN_LENGTH               ; total message length in bytes
                                    ; (include MN_SIZE in the length)
    LABEL   MN_SIZE

        ENDC    ; EXEC_PORTS_I
```

exec/resident.i

```
        IFND    EXEC_RESIDENT_I
EXEC_RESIDENT_I SET     1
**
**      $Filename: exec/resident.i $
**      $Release: 2.04 $
**      $Revision: 36.8 $
**      $Date: 90/11/01 $
**
**      Resident/ROMTag stuff.  Used to identify and initialize code modules.
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

    IFND EXEC_TYPES_I
    INCLUDE "exec/types.i"
    ENDC ; EXEC_TYPES_I


*----------------------------------------------------------------
*
*   Resident Module Tag
*
*----------------------------------------------------------------

 STRUCTURE RT,0
    UWORD RT_MATCHWORD                  ; word to match on (ILLEGAL)
    APTR  RT_MATCHTAG                   ; pointer to the above (RT_MATCHWORD)
    APTR  RT_ENDSKIP                    ; address to continue scan
    UBYTE RT_FLAGS                      ; various tag flags
    UBYTE RT_VERSION                    ; release version number
    UBYTE RT_TYPE                       ; type of module (NT_XXXXXX)
    BYTE  RT_PRI                        ; initialization priority
    APTR  RT_NAME                       ; pointer to node name
    APTR  RT_IDSTRING                   ; pointer to identification string
    APTR  RT_INIT                       ; pointer to init code
    LABEL RT_SIZE


;------ Match word definition:

RTC_MATCHWORD   EQU     $4AFC             ; The 68000 "ILLEGAL" instruction


;------ RT_FLAGS bit and field definitions:

    BITDEF RT,COLDSTART,0
    BITDEF RT,SINGLETASK,1
    BITDEF RT,AFTERDOS,2
    BITDEF RT,AUTOINIT,7                ; RT_INIT points to data structure

; Compatibility: (obsolete)
;RTM_WHEN       EQU     1               ; field position in RT_FLAGS
RTW_NEVER       EQU     0               ; never ever init
RTW_COLDSTART   EQU     1               ; init at coldstart time

        ENDC    ; EXEC_RESIDENT_I
```

---

exec/semaphores.i

```
        IFND    EXEC_SEMAPHORES_I
EXEC_SEMAPHORES_I       SET     1
**
**      $Filename: exec/semaphores.i $
**      $Release: 2.04 $
**      $Revision: 36.6 $
**      $Date: 90/05/10 $
**
**      Definitions for locking functions.
**
**      (C) Copyright 1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

    IFND EXEC_NODES_I
    INCLUDE "exec/nodes.i"
    ENDC        ; EXEC_NODES_I

    IFND EXEC_LISTS_I
    INCLUDE "exec/lists.i"
    ENDC        ; EXEC_LISTS_I

    IFND EXEC_PORTS_I
    INCLUDE "exec/ports.i"
    ENDC        ; EXEC_PORTS_I


*----------------------------------------------------------------
*
*   Signal Semaphore Structure
*
*----------------------------------------------------------------

** Private structure used by ObtainSemaphore()
 STRUCTURE   SSR,MLN_SIZE
    APTR     SSR_WAITER
    LABEL    SSR_SIZE

** Signal Semaphore data structure
 STRUCTURE   SS,LN_SIZE
    WORD     SS_NESTCOUNT
    STRUCT   SS_WAITQUEUE,MLH_SIZE
    STRUCT   SS_MULTIPLELINK,SSR_SIZE
    APTR     SS_OWNER
    WORD     SS_QUEUECOUNT
    LABEL    SS_SIZE


*----------------------------------------------------------------
*
*   Semaphore Structure (Procure/Vacate type)
*
*----------------------------------------------------------------


 STRUCTURE   SM,MP_SIZE
    WORD     SM_BIDS            ; number of bids for lock
    LABEL    SM_SIZE

*------ unions:

SM_LOCKMSG    EQU  MP_SIGTASK


        ENDC    ; EXEC_SEMAPHORES_I
```

exec/strings.i

Page 1

```
        IFND    EXEC_STRINGS_I
EXEC_STRINGS_I  SET     1
**
**      $Filename: exec/strings.i $
**      $Release: 2.04 $
**      $Revision: 36.5 $
**      $Date: 90/05/10 $
**
**      Macros for defining old style CR/LF terminated string constants
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

*------ Terminal Control:

EOS     EQU 0
BELL    EQU 7
LF      EQU 10
CR      EQU 13
BS      EQU 8
DEL     EQU $7F
NL      EQU LF


*---------------------------------------------------------------
*
*   String Support Macros
*
*---------------------------------------------------------------

STRING  MACRO
        dc.b    \1
        dc.b    0
        CNOP    0,2
        ENDM


STRINGL MACRO
        dc.b    13,10
        dc.b    \1
        dc.b    0
        CNOP    0,2
        ENDM


STRINGR MACRO
        dc.b    \1
        dc.b    13,10,0
        CNOP    0,2
        ENDM


STRINGLR MACRO
        dc.b    13,10
        dc.b    \1
        dc.b    13,10,0
        CNOP    0,2
        ENDM

        ENDC    ; EXEC_STRINGS_I
```

exec/tasks.i

Page 1

```
        IFND    EXEC_TASKS_I
EXEC_TASKS_I    SET     1
**
**      $Filename: exec/tasks.i $
**      $Release: 2.04 $
**      $Revision: 36.8 $
**      $Date: 90/10/22 $
**
**      Task Control Block, Signals, and Task flags.
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

    IFND EXEC_NODES_I
    INCLUDE "exec/nodes.i"
    ENDC        ; EXEC_NODES_I

    IFND EXEC_LISTS_I
    INCLUDE "exec/lists.i"
    ENDC        ; EXEC_LISTS_I

    IFND EXEC_PORTS_I
    INCLUDE "exec/ports.i"
    ENDC        ; EXEC_PORTS_I


*---------------------------------------------------------------
*
*       Task Control Structure
*
*       Please use Exec functions to modify task structure fields,
*       where available.
*
*---------------------------------------------------------------

 STRUCTURE  TC_Struct,LN_SIZE       ; was "TC"
    UBYTE   TC_FLAGS
    UBYTE   TC_STATE
    BYTE    TC_IDNESTCNT            ; intr disabled nesting
    BYTE    TC_TDNESTCNT            ; task disabled nesting
    ULONG   TC_SIGALLOC     ; sigs allocated
    ULONG   TC_SIGWAIT              ; sigs we are waiting for
    ULONG   TC_SIGRECVD     ; sigs we have received
    ULONG   TC_SIGEXCEPT            ; sigs we take as exceptions
    ;* Pointer to an extended task structure.  This structure is allocated
    ;* by V36 Exec if the proper flags in tc_ETaskFlags are set.  This
    ;* field was formerly defined as:
    ;*          UWORD   TC_TRAPALLOC        ; traps allocated
    ;*          UWORD   TC_TRAPABLE         ; traps enabled
    ;* Please see the Exec AllocTrap() and FreeTrap() calls.
    ;*
    APTR    tc_ETask                ; pointer to extended task structure
    APTR    TC_EXCEPTDATA           ; data for except proc
    APTR    TC_EXCEPTCODE           ; exception procedure
    APTR    TC_TRAPDATA     ; data for proc trap proc
    APTR    TC_TRAPCODE     ; proc trap procedure
    APTR    TC_SPREG                ; stack pointer
    APTR    TC_SPLOWER              ; stack lower bound
    APTR    TC_SPUPPER              ; stack upper bound + 2
    FPTR    TC_SWITCH               ; task losing CPU (function pointer)
    FPTR    TC_LAUNCH               ; task getting CPU (function pointer)
    STRUCT  TC_MEMENTRY,LH_SIZE     ; Allocated memory list.  Freed by RemTask()
    APTR    TC_Userdata             ; For use by the task; no restrictions!
    LABEL   TC_SIZE

```

exec/tasks.i Page 2

```

;Don't even _think_ about allocating one of these yourself.
 STRUCTURE      ETask,MN_SIZE
                APTR    et_Parent               ;Pointer to task (TC)
                ULONG   et_UniqueID             ;ID unique to this task
                STRUCT  et_Children,MLH_SIZE    ;List of children
                UWORD   et_TRAPALLOC
                UWORD   et_TRAPABLE
                ULONG   et_Result1              ;First result
                APTR    et_Result2              ;Result data pointer (AllocVec)
                STRUCT  et_TaskMsgPort,MP_SIZE
                LABEL   ETask_SIZEOF    ;_never_ depend on this size!


CHILD_NOTNEW    EQU     1       ;function not called from a new style task
CHILD_NOTFOUND  EQU     2       ;child not found
CHILD_EXITED    EQU     3       ;child has exited
CHILD_ACTIVE    EQU     4       ;child has exited


;Stack swap structure as passed to StackSwap()
;
 STRUCTURE      StackSwapStruct,0
                APTR    stk_Lower       ;Lowest byte of stack
                ULONG   stk_Upper       ;Upper end of stack (size + Lowest)
                APTR    stk_Pointer     ;Stack pointer at switch point
                LABEL   StackSwapStruct_SIZEOF


;------ TC_FLAGS Bits:

    BITDEF  T,PROCTIME,0
    BITDEF  T,ETASK,3
    BITDEF  T,STACKCHK,4
    BITDEF  T,EXCEPT,5
    BITDEF  T,SWITCH,6          ;Enable TC_SWITCH point
    BITDEF  T,LAUNCH,7          ;Enable TC_LAUNCH point


;------ Task States:

TS_INVALID  EQU     0
TS_ADDED    EQU     TS_INVALID+1
TS_RUN      EQU     TS_ADDED+1
TS_READY    EQU     TS_RUN+1
TS_WAIT     EQU     TS_READY+1
TS_EXCEPT   EQU     TS_WAIT+1
TS_REMOVED  EQU     TS_EXCEPT+1


;------ System Task Signals:

        BITDEF  SIG,ABORT,0
        BITDEF  SIG,CHILD,1
        BITDEF  SIG,BLIT,4      ; Note: same as SINGLE
        BITDEF  SIG,SINGLE,4    ; "single-threaded".  Note: same as BLIT
        BITDEF  SIG,INTUITION,5
        BITDEF  SIG,DOS,8


SYS_SIGALLOC    EQU     $0FFFF          ; pre-allocated signals
SYS_TRAPALLOC   EQU     $08000          ; pre-allocated traps

        ENDC    ; EXEC_TASKS_I
```

exec/types.i Page 1

```
        IFND    EXEC_TYPES_I
EXEC_TYPES_I    SET     1
**
**      $Filename: exec/types.i $
**      $Release: 2.04 $
**      $Revision: 36.9 $
**      $Date: 90/05/10 $
**
**      Data storage macros.  Must be included before any other Amiga include.
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**


INCLUDE_VERSION EQU 36  ;Version of the include files in use. (Do not
                        ;use this label for OpenLibrary() calls!)


EXTERN_LIB  MACRO
            XREF    _LVO\1
            ENDM

**
** Structure Building Macros
**
STRUCTURE   MACRO                   ; structure name, initial offset
\1          EQU     0
SOFFSET     SET     \2
            ENDM

FPTR        MACRO                   ; function pointer (32 bits - all bits valid)
\1          EQU     SOFFSET
SOFFSET     SET     SOFFSET+4
            ENDM

BOOL        MACRO                   ; boolean (16 bits)
\1          EQU     SOFFSET
SOFFSET     SET     SOFFSET+2
            ENDM

BYTE        MACRO                   ; byte (8 bits)
\1          EQU     SOFFSET
SOFFSET     SET     SOFFSET+1
            ENDM

UBYTE       MACRO                   ; unsigned byte (8 bits)
\1          EQU     SOFFSET
SOFFSET     SET     SOFFSET+1
            ENDM

WORD        MACRO                   ; word (16 bits)
\1          EQU     SOFFSET
SOFFSET     SET     SOFFSET+2
            ENDM

UWORD       MACRO                   ; unsigned word (16 bits)
\1          EQU     SOFFSET
SOFFSET     SET     SOFFSET+2
            ENDM

SHORT       MACRO                   ; obsolete - use WORD
\1          EQU     SOFFSET
SOFFSET     SET     SOFFSET+2
            ENDM

```

```
USHORT      MACRO                   ; obsolete - use UWORD
\1          EQU     SOFFSET
SOFFSET     SET     SOFFSET+2
            ENDM

LONG        MACRO                   ; long (32 bits)
\1          EQU     SOFFSET
SOFFSET     SET     SOFFSET+4
            ENDM

ULONG       MACRO                   ; unsigned long (32 bits)
\1          EQU     SOFFSET
SOFFSET     SET     SOFFSET+4
            ENDM

FLOAT       MACRO                   ; C float (32 bits)
\1          EQU     SOFFSET
SOFFSET     SET     SOFFSET+4
            ENDM

DOUBLE      MACRO                   ; C double (64 bits)
\1          EQU     SOFFSET
SOFFSET     SET     SOFFSET+8
            ENDM

APTR        MACRO                   ; untyped pointer (32 bits - all bits valid)
\1          EQU     SOFFSET
SOFFSET     SET     SOFFSET+4
            ENDM

CPTR        MACRO                   ; obsolete
\1          EQU     SOFFSET
SOFFSET     SET     SOFFSET+4
            ENDM

RPTR        MACRO                   ; signed relative pointer (16 bits)
\1          EQU     SOFFSET
SOFFSET     SET     SOFFSET+2
            ENDM

LABEL       MACRO                   ; Define a label without bumping the offset
\1          EQU     SOFFSET
            ENDM

STRUCT      MACRO                   ; Define a sub-structure
\1          EQU     SOFFSET
SOFFSET     SET     SOFFSET+\2
            ENDM

ALIGNWORD   MACRO                   ; Align structure offset to nearest word
SOFFSET     SET     (SOFFSET+1)&$fffffffe
            ENDM

ALIGNLONG   MACRO                   ; Align structure offset to nearest longword
SOFFSET     SET     (SOFFSET+3)&$fffffffc
            ENDM

**
** Enumerated variables.  Use ENUM to set a base number, and EITEM to assign
** incrementing values.  ENUM can be used to set a new base at any time.
**
ENUM        MACRO   ;[new base]
            IFC     '\1',''
EOFFSET     SET     0               ; Default to zero
            ENDC
            IFNC    '\1',''
```

```
EOFFSET     SET     \1
            ENDC
            ENDM

EITEM       MACRO   ;label
\1          EQU     EOFFSET
EOFFSET     SET     EOFFSET+1
            ENDM

**
**  Bit Definition Macro
**
**  Given:
**      BITDEF  MEM,CLEAR,16
**
**  Yields:
**      MEMB_CLEAR  EQU 16                      ; Bit number
**      MEMF_CLEAR  EQU 1<<16                   ; Bit mask
**

BITDEF      MACRO   ; prefix,&name,&bitnum
            BITDEF0 \1,\2,B_,\3
\@BITDEF    SET     1<<\3
            BITDEF0 \1,\2,F_,\@BITDEF
            ENDM

BITDEF0     MACRO   ; prefix,&name,&type,&value
\1\3\2      EQU     \4
            ENDM

**
** LIBRARY_VERSION is now obsolete.  Please use LIBRARY_MINIMUM or code
** the specific minimum library version you require.
**              LIBRARY_VERSION EQU     36
**
LIBRARY_MINIMUM EQU 33  ;Lowest version supported by Commodore-Amiga

        ENDC    ; EXEC_TYPES_I
        IFND    GRAPHICS_CLIP_I
GRAPHICS_CLIP_I SET     1
**
**      $Filename: graphics/clip.i $
**      $Release: 2.04 $
**      $Revision: 37.1 $
**      $Date: 91/01/28 $
**
**
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

    IFND    EXEC_TYPES_I
    include 'exec/types.i'
    ENDC

    IFND    GRAPHICS_GFX_I
    include "graphics/gfx.i"
    ENDC

    IFND    EXEC_SEMAPHORES_I
    include "exec/semaphores.i"
    ENDC

NEWLOCKS        equ     1

 STRUCTURE  Layer,0
    LONG    lr_front
    LONG    lr_back
    LONG    lr_ClipRect
    LONG    lr_rp
    WORD    lr_MinX
    WORD    lr_MinY
    WORD    lr_MaxX
    WORD    lr_MaxY
        STRUCT  lr_reserved,4
        WORD    lr_priority
    WORD    lr_Flags
    LONG    lr_SuperBitMap
    LONG    lr_SuperClipRect
    APTR    lr_Window
    WORD    lr_Scroll_X
    WORD    lr_Scroll_Y
    APTR    lr_cr
    APTR    lr_cr2
    APTR    lr_crnew
    APTR    lr_SuperSaverClipRects
    APTR    lr__cliprects
    APTR    lr_LayerInfo
*                       just by lucky coincidence
*                       this is not confused with simplesprites
        STRUCT  lr_Lock,SS_SIZE
        APTR    lr_BackFill
        ULONG   lr_reserved1
        APTR    lr_ClipRegion
        APTR    lr_saveClipRects
        STRUCT  lr_reserved2,22
    APTR    lr_DamageList
    LABEL   lr_SIZEOF

 STRUCTURE  ClipRect,0
    LONG    cr_Next
    LONG    cr_prev
    LONG    cr_lobs
```

```
    LONG    cr_BitMap
    WORD    cr_MinX
    WORD    cr_MinY
    WORD    cr_MaxX
    WORD    cr_MaxY
    APTR    cr__p1
    APTR    cr__p2
    LONG    cr_reserved
    LONG    cr_Flags
 LABEL      cr_SIZEOF

* internal cliprect flags
CR_NEEDS_NO_CONCEALED_RASTERS   equ     1
CR_NEEDS_NO_LAYERBLIT_DAMAGE    equ     2

* defines for clipping
ISLESSX equ 1
ISLESSY equ 2
ISGRTRX equ 4
ISGRTRY equ 8

* for ancient history reasons
        IFND    lr_Front
lr_Front        equ lr_front
lr_Back equ lr_back
lr_RastPort     equ     lr_rp
cr_Prev equ cr_prev
cr_LObs equ     cr_lobs
        ENDC

        ENDC    ; GRAPHICS_CLIP_I
```

graphics/copper.i Page 1

```
   IFND  GRAPHICS_COPPER_I
GRAPHICS_COPPER_I SET   1
**
**      $Filename: graphics/copper.i $
**      $Release: 2.04 $
**      $Revision: 37.1 $
**      $Date: 91/02/12 $
**
**      graphics copper list intruction definitions
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

    IFND    EXEC_TYPES_I
    include 'exec/types.i'
    ENDC

COPPER_MOVE equ 0       * pseude opcode for move #XXXX,dir
COPPER_WAIT equ 1       * pseudo opcode for wait y,x
CPRNXTBUF   equ 2       * continue processing with next buffer
CPR_NT_LOF  equ $8000   * copper instruction only for short frames
CPR_NT_SHT  equ $4000   * copper instruction only for long frames
CPR_NT_SYS  equ $2000   * copper user instruction only

   STRUCTURE   CopIns,0
      WORD  ci_OpCode         * 0 = move, 1 = wait
      STRUCT   ci_nxtlist,0   * UNION
      STRUCT   ci_VWaitPos,0
      STRUCT   ci_DestAddr,2

      STRUCT   ci_HWaitPos,0
      STRUCT   ci_DestData,2

   LABEL ci_SIZEOF

* structure of cprlist that points to list that hardware actually executes
   STRUCTURE   cprlist,0
      APTR  crl_Next
      APTR  crl_start
      WORD  crl_MaxCount
   LABEL crl_SIZEOF

   STRUCTURE   CopList,0
      APTR  cl_Next        * next block for this copper list
      APTR  cl__CopList    * system use
      APTR  cl__ViewPort   * system use
      APTR  cl_CopIns      * start of this block
      APTR  cl_CopPtr      * intermediate ptr
      APTR  cl_CopLStart   * mrgcop fills this in for Long Frame
      APTR  cl_CopSStart   * mrgcop fills this in for Short Frame
      WORD  cl_Count       * intermediate counter
      WORD  cl_MaxCount    * max # of copins for this block
      WORD  cl_DyOffset    * offset this copper list vertical waits
   LABEL cl_SIZEOF

   STRUCTURE   UCopList,0
      APTR     ucl_Next
      APTR     ucl_FirstCopList * head node of this copper list
      APTR     ucl_CopList      * node in use
   LABEL ucl_SIZEOF

*  private graphics data structure
   STRUCTURE   copinit,0
        STRUCT  copinit_vsync_hblank,4
        STRUCT  copinit_diwstart,8
```

graphics/copper.i Page 2

```
        STRUCT  copinit_diagstrt,8
        STRUCT  copinit_sprstrtup,2*(2*8*2)
        STRUCT  copinit_wait14,2*(2+2)
        STRUCT  copinit_genloc,2*(4+(2*2)+2)
        STRUCT  copinit_sprstop,8
   LABEL copinit_SIZEOF

   ENDC ; GRAPHICS_COPPER_I
```

graphics/display.i

```
        IFND    GRAPHICS_DISPLAY_I
GRAPHICS_DISPLAY_I      SET     1
**
**      $Filename: graphics/display.i $
**      $Release: 2.04 $
**      $Revision: 37.0 $
**      $Date: 91/01/07 $
**
**      include define file for display control registers
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

* bplcon0 defines
MODE_640     equ    $8000
PLNCNTMSK    equ    $7              * how many bit planes?
*                                   * 0 = none, 1->6 = 1->6, 7 = reserved
PLNCNTSHFT   equ    12              * bits to shift for bplcon0
PF2PRI       equ    $40             * bplcon2 bit
COLORON      equ    $0200     * disable color burst
DBLPF        equ    $400
HOLDNMODIFY  equ    $800
INTERLACE    equ    4               * interlace mode for 400

* bplcon1 defines
PFA_FINE_SCROLL       equ    $F
PFB_FINE_SCROLL_SHIFT equ    4
PF_FINE_SCROLL_MASK   equ    $F

* display window start and stop defines
DIW_HORIZ_POS         equ    $7F    * horizontal start/stop
DIW_VRTCL_POS         equ    $1FF   * vertical start/stop
DIW_VRTCL_POS_SHIFT   equ    7

* Data fetch start/stop horizontal position
DFTCH_MASK            equ    $FF

* vposr bits
VPOSRLOF              equ    $8000

        ENDC    ; GRAPHICS_DISPLAY_I
```

graphics/displayinfo.i

```
        IFND    GRAPHICS_DISPLAYINFO_I
GRAPHICS_DISPLAYINFO_I  SET     1
**
**      $Filename: graphics/displayinfo.i $
**      $Release: 2.04 $
**      $Revision: 37.4 $
**      $Date: 91/02/12 $
**
**      include define file for display control registers
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

    IFND        EXEC_TYPES_I
    include 'exec/types.i'
    ENDC

    IFND        GRAPHICS_GFX_I
    include 'graphics/gfx.i'
    ENDC

    IFND        GRAPHICS_MONITOR_I
    include 'graphics/monitor.i'
    ENDC

    IFND        UTILITY_TAGITEM_I
    include 'utility/tagitem.i'
    ENDC

* datachunk type identifiers

DTAG_DISP       equ             $80000000
DTAG_DIMS       equ             $80001000
DTAG_MNTR       equ             $80002000
DTAG_NAME       equ             $80003000


    STRUCTURE   QueryHeader,0
        ULONG   qh_StructID      ; datachunk type identifier
        ULONG   qh_DisplayID     ; copy of display record key
        ULONG   qh_SkipID        ; TAG_SKIP -- see tagitems.h
        ULONG   qh_Length        ; length of data in double-longwords
    LABEL qh_SIZEOF

    STRUCTURE   DisplayInfo,qh_SIZEOF
        UWORD   dis_NotAvailable         ; if NULL available, else see defines
        ULONG   dis_PropertyFlags        ; Properties of this mode see defines
        STRUCT  dis_Resolution,tpt_SIZEOF ; ticks-per-pixel X/Y
        UWORD   dis_PixelSpeed           ; aproximation in nanoseconds
        UWORD   dis_NumStdSprites        ; number of standard amiga sprites
        UWORD   dis_PaletteRange         ; distinguishable shades available
        STRUCT  dis_SpriteResolution,tpt_SIZEOF ; sprite ticks-per-pixel X/Y
        STRUCT  dis_pad,4
        STRUCT  dis_reserved,8           ; terminator
    LABEL dis_SIZEOF

* availability

DI_AVAIL_NOCHIPS        equ     $0001
DI_AVAIL_NOMONITOR      equ     $0002
DI_AVAIL_NOTWITHGENLOCK equ     $0004

* mode properties

DIPF_IS_LACE            equ     $00000001
```

graphics/displayinfo.i Page 2

```
DIPF_IS_DUALPF          equ     $00000002
DIPF_IS_PF2PRI          equ     $00000004
DIPF_IS_HAM             equ     $00000008

DIPF_IS_ECS             equ     $00000010       ;*      note: ECS modes (SHIRES,
  VGA, and **
*                                                       ;*      PRODUCTI
  VITY) do not support        **
*                                                       ;*      attached
   sprites.                   **
DIPF_IS_PAL             equ     $00000020
DIPF_IS_SPRITES         equ     $00000040
DIPF_IS_GENLOCK         equ     $00000080

DIPF_IS_WB              equ     $00000100
DIPF_IS_DRAGGABLE       equ     $00000200
DIPF_IS_PANELLED        equ     $00000400
DIPF_IS_BEAMSYNC        equ     $00000800

DIPF_IS_EXTRAHALFBRITE equu     $00001000

    STRUCTURE DimensionInfo,qh_SIZEOF
        UWORD   dim_MaxDepth            ; log2( max number of colors
        UWORD   dim_MinRasterWidth      ; minimum width in pixels
        UWORD   dim_MinRasterHeight     ; minimum height in pixels
        UWORD   dim_MaxRasterWidth      ; maximum width in pixels
        UWORD   dim_MaxRasterHeight     ; maximum height in pixels
        STRUCT  dim_Nominal,ra_SIZEOF   ; "standard" dimensions
        STRUCT  dim_MaxOScan,ra_SIZEOF  ; fixed, hardware dependant
        STRUCT  dim_VideoOScan,ra_SIZEOF ; fixed, hardware dependant
        STRUCT  dim_TxtOScan,ra_SIZEOF  ; editable via preferences
        STRUCT  dim_StdOScan,ra_SIZEOF  ; editable via preferences
        STRUCT  dim_pad,14
        STRUCT  dim_reserved,8          ; terminator
    LABEL dim_SIZEOF

    STRUCTURE MonitorInfo,qh_SIZEOF
        APTR    mtr_Mspc                ; pointer to monitor specification
        STRUCT  mtr_ViewPosition,tpt_SIZEOF     ; editable via preferences
        STRUCT  mtr_ViewResolution,tpt_SIZEOF   ; monitor ticks-per-pixel
        STRUCT  mtr_ViewPositionRange,ra_SIZEOF ; fixed, hardware dependant
        UWORD   mtr_TotalRows           ; display height in scanlines
        UWORD   mtr_TotalColorClocks    ; scanline width in 280 ns units
        UWORD   mtr_MinRow              ; absolute minimum active scanline
        WORD    mtr_Compatibility       ; how this coexists with others
        STRUCT  mtr_pad,36
        STRUCT  mtr_reserved,8          ; terminator
    LABEL mtr_SIZEOF

* monitor compatibility

MCOMPAT_MIXED           equ  0  ; can share display with other MCOMPAT_MIXED
MCOMPAT_SELF            equ  1  ; can share only within same monitor
MCOMPAT_NOBODY          equ -1  ; only one viewport at a time

DISPLAYNAMELEN          equ 32

    STRUCTURE NameInfo,qh_SIZEOF
        STRUCT  nif_Name,DISPLAYNAMELEN
        STRUCT  nif_reserved,8          ; terminator
    LABEL nif_SIZEOF

* DisplayInfoRecord identifiers

INVALID_ID              equ     -1

```

graphics/displayinfo.i Page 3

```
*normal identifiers

MONITOR_ID_MASK         equ     $FFFF1000

DEFAULT_MONITOR_ID      equ     $00000000
NTSC_MONITOR_ID         equ     $00011000
PAL_MONITOR_ID          equ     $00021000

* the following 20 composite keys are for Modes on the default Monitor
* ntsc & pal "flavors" of these particular keys may be made by or'ing
* the ntsc or pal MONITOR_ID with the desired MODE_KEY...

LORES_KEY               equ     $00000000
HIRES_KEY               equ     $00008000
SUPER_KEY               equ     $00008020
HAM_KEY                 equ     $00000800
LORESLACE_KEY           equ     $00000004
HIRESLACE_KEY           equ     $00008004
SUPERLACE_KEY           equ     $00008024
HAMLACE_KEY             equ     $00000804
LORESDPF_KEY            equ     $00000400
HIRESDPF_KEY            equ     $00008400
SUPERDPF_KEY            equ     $00008420
LORESLACEDPF_KEY        equ     $00000404
HIRESLACEDPF_KEY        equ     $00008404
SUPERLACEDPF_KEY        equ     $00008424
LORESDPF2_KEY           equ     $00000440
HIRESDPF2_KEY           equ     $00008440
SUPERDPF2_KEY           equ     $00008460
LORESLACEDPF2_KEY       equ     $00000444
HIRESLACEDPF2_KEY       equ     $00008444
SUPERLACEDPF2_KEY       equ     $00008464
EXTRAHALFBRITE_KEY      equ     $00000080
EXTRAHALFBRITELACE_KEY  equ     $00000084

* vga identifiers

VGA_MONITOR_ID          equ     $00031000

VGAEXTRALORES_KEY       equ     $00031004
VGALORES_KEY            equ     $00039004
VGAPRODUCT_KEY          equ     $00039024
VGAHAM_KEY              equ     $00031804
VGAEXTRALORESLACE_KEY   equ     $00031005
VGALORESLACE_KEY        equ     $00039005
VGAPRODUCTLACE_KEY      equ     $00039025
VGAHAMLACE_KEY          equ     $00031805
VGAEXTRALORESDPF_KEY    equ     $00031404
VGALORESDPF_KEY         equ     $00039404
VGAPRODUCTDPF_KEY       equ     $00039424
VGAEXTRALORESLACEDPF_KEY equ    $00031405
VGALORESLACEDPF_KEY     equ     $00039405
VGAPRODUCTLACEDPF_KEY   equ     $00039425
VGAEXTRALORESDPF2_KEY   equ     $00031444
VGALORESDPF2_KEY        equ     $00039444
VGAPRODUCTDPF2_KEY      equ     $00039464
VGAEXTRALORESLACEDPF2_KEY equ   $00031445
VGALORESLACEDPF2_KEY    equ     $00039445
VGAPRODUCTLACEDPF2_KEY  equ     $00039465
VGAEXTRAHALFBRITE_KEY   equ     $00031084
VGAEXTRAHALFBRITELACE_KEY equ   $00031085

* a2024 identifiers

A2024_MONITOR_ID        equ     $00041000

```

graphics/displayinfo.i

```
A2024TENHERTZ_KEY       equ     $00041000
A2024FIFTEENHERTZ_KEY   equ     $00049000

* prototype identifiers

PROTO_MONITOR_ID        equ     $00051000

    ENDC        ; GRAPHICS_DISPLAYINFO_I
```

graphics/gels.i

```
        IFND    GRAPHICS_GELS_I
GRAPHICS_GELS_I SET     1
**
**      $Filename: graphics/gels.i $
**      $Release: 2.04 $
**      $Revision: 37.0 $
**      $Date: 91/01/07 $
**
**      include file for AMIGA GELS (Graphics Elements)
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

    IFND    EXEC_TYPES_I
    include 'exec/types.i'
    ENDC

*------ VS_vSflags ------------------------------------------------

*   ;-- user-set vSprite flags --
SUSERFLAGS  EQU $00FF           ; mask of all user-settable vSprite-flags
    BITDEF  VS,VSPRITE,0        ; set if vSprite, clear if bob
    BITDEF  VS,SAVEBACK,1       ; set if background is to be saved/restored
    BITDEF  VS,OVERLAY,2        ; set to mask image of bob onto background
    BITDEF  VS,MUSTDRAW,3       ; set if vSprite absolutely must be drawn
*   ;-- system-set vSprite flags --
    BITDEF  VS,BACKSAVED,8      ; this bob's background has been saved
    BITDEF  VS,BOBUPDATE,9      ; temporary flag, useless to outside world
    BITDEF  VS,GELGONE,10       ; set if gel is completely clipped (offscreen)
    BITDEF  VS,VSOVERFLOW,11    ; vSprite overflow (if MUSTDRAW set we draw!)


*------ B_flags ----------------------------------------------------
*   ;-- these are the user flag bits --
BUSERFLAGS  EQU $00FF           ; mask of all user-settable bob-flags
    BITDEF  B,SAVEBOB,0         ; set to not erase bob
    BITDEF  B,BOBISCOMP,1       ; set to identify bob as animComp
*   ;-- these are the system flag bits --
    BITDEF  B,BWAITING,8        ; set while bob is waiting on 'after'
    BITDEF  B,BDRAWN,9          ; set when bob is drawn this DrawG pass
    BITDEF  B,BOBSAWAY,10       ; set to initiate removal of bob
    BITDEF  B,BOBNIX,11         ; set when bob is completely removed
    BITDEF  B,SAVEPRESERVE,12 ; for back-restore during double-buffer
    BITDEF  B,OUTSTEP,13        ; for double-clearing if double-buffer


*------ defines for the animation procedures ------------------------

ANFRACSIZE  EQU 6
ANIMHALF    EQU $0020
RINGTRIGGER EQU $0001

*------ macros -----------------------------------------------------
* these are GEL functions that are currently simple enough to exist as a
* definition.  It should not be assumed that this will always be the case

InitAnimate MACRO   * &animKey
       CLR.L   \1
       ENDM


RemBob      MACRO   * &b
       OR.W    #BF_BOBSAWAY,b_BobFlags+\1
       ENDM

```

```
*------ VS : vSprite --------------------------------------------------
 STRUCTURE  VS,0    ; vSprite
*   -- SYSTEM VARIABLES --
*   GEL linked list forward/backward pointers sorted by y,x value
    APTR    vs_NextVSprite    ; struct *vSprite
    APTR    vs_PrevVSprite    ; struct *vSprite
*   GEL draw list constructed in the order the bobs are actually drawn, then
*   list is copied to clear list
*   must be here in vSprite for system boundary detection
    APTR    vs_DrawPath       ; struct *vSprite: pointer of overlay drawing
    APTR    vs_ClearPath      ; struct *vSprite: pointer for overlay clearing
*   the vSprite positions are defined in (y,x) order to make sorting
*   sorting easier, since (y,x) as a long integer
    WORD    vs_Oldy           ; previous position
    WORD    vs_Oldx           ;
*   -- COMMON VARIABLES --
    WORD    vs_VSFlags        ; vSprite flags
*   -- USER VARIABLES --
*    the vSprite positions are defined in (y,x) order to make sorting
*    easier, since (y,x) as a long integer
    WORD    vs_Y              ; screen position
    WORD    vs_X
    WORD    vs_Height
    WORD    vs_Width          ; number of words per row of image data
    WORD    vs_Depth          ; number of planes of data
    WORD    vs_MeMask         ; which types can collide with this vSprite
    WORD    vs_HitMask        ; which types this vSprite can collide with
    APTR    vs_ImageData      ; *WORD pointer to vSprite image
*    borderLine is the one-dimensional logical OR of all
*    the vSprite bits, used for fast collision detection of edge
    APTR    vs_BorderLine     ; *WORD: logical OR of all vSprite bits
    APTR    vs_CollMask       ; *WORD: similar to above except this is a
*    matrix pointer to this vSprite's color definitions (not used by bobs)
    APTR    vs_SprColors      ; *WORD
    APTR    vs_VSBob          ; struct *bob: points home if this vSprite is
                              ;  part of a bob
*    planePick flag:  set bit selects a plane from image, clear bit selects
*         use of shadow mask for that plane
*    OnOff flag: if using shadow mask to fill plane, this bit (corresponding
*         to bit in planePick) describes whether to fill with 0's or 1's
*    There are two uses for these flags:
*              - if this is the vSprite of a bob, these flags describe how
*                the bob is to be drawn into memory
*              - if this is a simple vSprite and the user intends on setting
*                the MUSTDRAW flag of the vSprite, these flags must be set
*                too to describe which color registers the user wants for
*                the image
    BYTE    vs_PlanePick
    BYTE    vs_PlaneOnOff
    LABEL   vs_SUserExt       ; user definable
    LABEL   vs_SIZEOF


*------ BOB : bob ----------------------------------------------------

 STRUCTURE  BOB,0     ; bob: blitter object
*   -- COMMON VARIABLES --
    WORD    bob_BobFlags      ; general purpose flags (see definitions below)
*   -- USER VARIABLES --
    APTR    bob_SaveBuffer    ; *WORD pointer to the buffer for background
*    save used by bobs for "cookie-cutting" and multi-plane masking
    APTR    bob_ImageShadow   ; *WORD
*    pointer to BOBs for sequenced drawing of bobs
*      for correct overlaying of multiple component animations
    APTR    bob_Before        ; struct *bob: draw this bob before bob pointed
                              ; to by before
```

```
    APTR    bob_After         ; struct *bob: draw this bob after bob pointed
                              ; to by after
    APTR    bob_BobVSprite    ; struct *vSprite: this bob's vSprite definition
    APTR    bob_BobComp       ; struct *animComp: pointer to this bob's
                              ; animComp def
    APTR    bob_DBuffer       ; struct dBufPacket: pointer to this bob's
                              ; dBuf packet
    LABEL   bob_BUserExt      ; bob user extension
    LABEL   bob_SIZEOF

*------ AC : animComp -------------------------------------------------

 STRUCTURE  AC,0    ; animComp
*   -- COMMON VARIABLES --
    WORD    ac_CompFlags      ; animComp flags for system & user
*    timer defines how long to keep this component active:
*      if set non-zero, timer decrements to zero then switches to nextSeq
*      if set to zero, animComp never switches
    WORD    ac_Timer
*   -- USER VARIABLES --
*    initial value for timer when the animComp is activated by the system
    WORD    ac_TimeSet
*    pointer to next and previous components of animation object
    APTR    ac_NextComp       ; struct *animComp
    APTR    ac_PrevComp       ; struct *animComp
*    pointer to component component definition of next image in sequence
    APTR    ac_NextSeq        ; struct *animComp
    APTR    ac_PrevSeq        ; struct *animComp
    APTR    ac_AnimCRoutine   ; address of special animation procedure
    WORD    ac_YTrans         ; initial y translation (if this is a component)
    WORD    ac_XTrans         ; initial x translation (if this is a component)
    APTR    ac_HeadOb         ; struct *animOb
    APTR    ac_AnimBob        ; struct *bob
    LABEL   ac_SIZE

*------ AO : animOb ---------------------------------------------------

 STRUCTURE  AO,0    ; animOb
*   -- SYSTEM VARIABLES --
    APTR    ao_NextOb         ; struct *animOb
    APTR    ao_PrevOb         ; struct *animOb
*    number of calls to Animate this animOb has endured
    LONG    ao_Clock
    WORD    ao_AnOldY         ; old y,x coordinates
    WORD    ao_AnOldX         ;
*   -- COMMON VARIABLES --
    WORD    ao_AnY            ; y,x coordinates of the animOb
    WORD    ao_AnX            ;
*   -- USER VARIABLES --
    WORD    ao_YVel           ; velocities of this object
    WORD    ao_XVel           ;
    WORD    ao_XAccel         ; accelerations of this object
    WORD    ao_YAccel         ;   !!! backwards !!!
    WORD    ao_RingYTrans     ; ring translation values
    WORD    ao_RingXTrans     ;
    APTR    ao_AnimORoutine   ; address of special animation procedure
    APTR    ao_HeadComp       ; struct *animComp: pointer to first component
    LABEL   ao_AUserExt       ; animOb user extension
    LABEL   ao_SIZEOF


*------ DBP : dBufPacket ---------------------------------------------
* dBufPacket defines the values needed to be saved across buffer to buffer
*   when in double-buffer mode

 STRUCTURE  DBP,0             ; dBufPacket
```

graphics/gels.i

```
    WORD    dbp_BufY         ; save the other buffers screen coordinates
    WORD    dbp_BufX         ;
    APTR    dbp_BufPath      ; struct *vSprite: carry the draw path over
                             ; the gap
*    these pointers must be filled in by the user
*    pointer to other buffer's background save buffer
    APTR    dbp_BufBuffer    ; *WORD
*    pointer to other buffer's background plane pointers
    APTR    dbp_BufPlanes    ; **WORD
    LABEL   dbp_SIZEOF

        ENDC    ; GRAPHICS_GELS_I
```

graphics/gfx.i

```
        IFND    GRAPHICS_GFX_I
GRAPHICS_GFX_I  SET     1
**
**      $Filename: graphics/gfx.i $
**      $Release: 2.04 $
**      $Revision: 37.0 $
**      $Date: 91/01/07 $
**
**
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

    IFND    EXEC_TYPES_I
    include 'exec/types.i'
    ENDC

BITSET      equ $8000
BITCLR      equ 0
AGNUS       equ 1
DENISE      equ 1

    STRUCTURE   BitMap,0
    WORD    bm_BytesPerRow
    WORD    bm_Rows
    BYTE    bm_Flags
    BYTE    bm_Depth
    WORD    bm_Pad
    STRUCT  bm_Planes,8*4
    LABEL   bm_SIZEOF

   STRUCTURE   Rectangle,0
      WORD  ra_MinX
      WORD  ra_MinY
      WORD  ra_MaxX
      WORD  ra_MaxY
   LABEL    ra_SIZEOF

   STRUCTURE   Rect32,0
      LONG  r32_MinX
      LONG  r32_MinY
      LONG  r32_MaxX
      LONG  r32_MaxY
   LABEL    r32_SIZEOF

   STRUCTURE   tPoint,0
      WORD  tpt_x
      WORD  tpt_y
   LABEL    tpt_SIZEOF

    ENDC        ; GRAPHICS_GFX_I
```

graphics/gfxbase.i Page 1

```
    IFND     GRAPHICS_GFXBASE_I
GRAPHICS_GFXBASE_I  SET 1
**
**      $Filename: graphics/gfxbase.i $
**      $Release: 2.04 $
**      $Revision: 37.2 $
**      $Date: 91/02/07 $
**
**      graphics base definitions
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

    IFND     EXEC_LISTS_I
    include 'exec/lists.i'
    ENDC
    IFND     EXEC_LIBRARIES_I
    include 'exec/libraries.i'
    ENDC
    IFND     EXEC_INTERRUPTS_I
    include 'exec/interrupts.i'
    ENDC

 STRUCTURE   GfxBase,LIB_SIZE
    APTR     gb_ActiView     ; struct *View
    APTR     gb_copinit      ; struct *copinit; ptr to copper start up list
    APTR     gb_cia      ; for 6526 resource use
    APTR     gb_blitter      ; for blitter resource use
    APTR     gb_LOFlist      ; current copper list being run
    APTR     gb_SHFlist      ; current copper list being run
    APTR     gb_blthd        ; struct *bltnode
    APTR     gb_bltt1        ;
    APTR     gb_bsblthd      ;
    APTR     gb_bsbltt1      ;
    STRUCT   gb_vbsrv,IS_SIZE
    STRUCT   gb_timsrv,IS_SIZE
    STRUCT   gb_bltsrv,IS_SIZE
    STRUCT   gb_TextFonts,LH_SIZE
    APTR     gb_DefaultFont
    UWORD    gb_Modes        ; copy of bltcon0
    BYTE     gb_VBlank
    BYTE     gb_Debug
    UWORD    gb_BeamSync
    WORD     gb_system_bplcon0
    BYTE     gb_SpriteReserved
    BYTE     gb_bytereserved

    WORD     gb_Flags
    WORD     gb_BlitLock
        WORD     gb_BlitNest
        STRUCT   gb_BlitWaitQ,LH_SIZE
        APTR     gb_BlitOwner
        STRUCT   gb_TOF_WaitQ,LH_SIZE

        WORD     gb_DisplayFlags
        APTR     gb_SimpleSprites
        WORD     gb_MaxDisplayRow
        WORD     gb_MaxDisplayColumn
        WORD     gb_NormalDisplayRows
        WORD     gb_NormalDisplayColumns
        WORD     gb_NormalDPMX
        WORD     gb_NormalDPMY

        APTR     gb_LastChanceMemory
        APTR     gb_LCMptr
```

graphics/gfxbase.i Page 2

```

        WORD     gb_MicrosPerLine         ; usecs per line times 256
        WORD     gb_MinDisplayColumn

        UBYTE    gb_ChipRevBits0          ; agnus/denise new features
        STRUCT   gb_crb_reserved,5

        STRUCT   gb_monitor_id,2 ; normally null
        STRUCT   gb_hedley,4*8
        STRUCT   gb_hedley_sprites,4*8
        STRUCT   gb_hedley_sprites1,4*8
        WORD     gb_hedley_count
        WORD     gb_hedley_flags
        WORD     gb_hedley_tmp
        APTR     gb_hash_table
        UWORD    gb_current_tot_rows
        UWORD    gb_current_tot_cclks
        UBYTE    gb_hedley_hint
        UBYTE    gb_hedley_hint2
        STRUCT   gb_nreserved,4*4
        APTR     gb_a2024_sync_raster
        WORD     gb_control_delta_pal
        WORD     gb_control_delta_ntsc
        APTR     gb_current_monitor
        STRUCT   gb_MonitorList,LH_SIZE
        APTR     gb_default_monitor
        APTR     gb_MonitorListSemaphore
        APTR     gb_DisplayInfoDataBase
        APTR     gb_ActiViewCprSemaphore
        APTR     gb_UtilityBase
        APTR     gb_ExecBase
    LABEL    gb_SIZE

* bits for dalestuff, which may go away when blitter becomes a resource
OWNBLITTERn equ 0   * blitter owned bit
QBOWNERn    equ 1   * blitter owned by blit queuer

* flag bits for ChipRevBits
        BITDEF   GFX,BIG_BLITS,0
        BITDEF   GFX,HR_AGNUS,0
        BITDEF   GFX,HR_DENISE,1


QBOWNER     equ 1<<QBOWNERn

* flag bits for DisplayFlags

NTSCn            equ 0
NTSC             equ 1<<NTSCn

GENLOCn          equ 1
GENLOC           equ 1<<GENLOCn

PALn             equ 2
PAL              equ 1<<PALn

TODA_SAFEn       equ 3
TODA_SAFE        equ 1<<TODA_SAFEn

BLITMSG_FAULTn   equ 2
BLITMSG_FAULT    equ 1<<BLITMSG_FAULTn

* handy name macro

GRAPHICSNAME     MACRO
                 DC.B  'graphics.library',0
```

graphics/gfxbase.i

```
                ENDM

        ENDC        ; GRAPHICS_GFXBASE_I
```

graphics/gfxnodes.i

```
        IFND    GRAPHICS_GFXNODES_I
GRAPHICS_GFXNODES_I     EQU     1
**
**      $Filename: graphics/gfxnodes.i $
**      $Release: 2.04 $
**      $Revision: 37.0 $
**      $Date: 91/01/07 $
**
**      graphics extended node definitions
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

    IFND    EXEC_TYPES_I
    include 'exec/types.i'
    ENDC

        STRUCTURE       XLN,0
        APTR    XLN_SUCC
        APTR    XLN_PRED
        UBYTE   XLN_TYPE
        BYTE    XLN_PRI
        APTR    XLN_NAME
        UBYTE   XLN_SUBSYSTEM
        UBYTE   XLN_SUBTYPE
        LONG    XLN_LIBRARY
        LONG    XLN_INIT
        LABEL   XLN_SIZE

SS_GRAPHICS     EQU     $02

VIEW_EXTRA_TYPE         EQU     1
VIEWPORT_EXTRA_TYPE     EQU     2
SPECIAL_MONITOR_TYPE    EQU     3
MONITOR_SPEC_TYPE       EQU     4

        ENDC    ; GRAPHICS_GFXNODES_I
```

graphics/layers.i

```
        IFND    GRAPHICS_LAYERS_I
GRAPHICS_LAYERS_I       SET     1
**
**      $Filename: graphics/layers.i $
**      $Release: 2.04 $
**      $Revision: 37.0 $
**      $Date: 91/01/07 $
**
**
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

    IFND    EXEC_SEMAPHORES_I
       include "exec/semaphores.i"
    ENDC

    IFND  EXEC_LISTS_I
       include  "exec/lists.i"
    ENDC

* these should be clip.i/h but you know backwards compatibility etc.
LAYERSIMPLE             equ   1
LAYERSMART              equ   2
LAYERSUPER              equ   4
LAYERUPDATING           equ   $10
LAYERBACKDROP           equ   $40
LAYERREFRESH            equ   $80
LAYER_CLIPRECTS_LOST    equ   $100

LMN_REGION  equ -1

    STRUCTURE  Layer_Info,0
    APTR       li_top_layer
    APTR       li_check_lp
    APTR       li_obs
    STRUCT     li_FreeClipRects,MLH_SIZE
    STRUCT     li_Lock,SS_SIZE
        STRUCT     li_gs_Head,LH_SIZE
        LONG            li_long_reserved
        WORD       li_Flags
    BYTE       li_fatten_count
        BYTE       li_LockLayersCount
    WORD       li_LayerInfo_extra_size
        APTR            li_blitbuff
    APTR       li_LayerInfo_extra
    LABEL      li_SIZEOF

NEWLAYERINFO_CALLED     equ 1
ALERTLAYERSNOMEM        equ $83010000

        ENDC    ; GRAPHICS_LAYERS_I
```

graphics/monitor.i

```
    IFND    GRAPHICS_MONITOR_I
GRAPHICS_MONITOR_I SET 1
**
**      $Filename: graphics/monitor.i $
**      $Release: 2.04 $
**      $Revision: 37.0 $
**      $Date: 91/01/07 $
**
**      graphics monitorspec definitions
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

    IFND    EXEC_TYPES_I
    include 'exec/types.i'
    ENDC

    IFND    GRAPHICS_GFX_I
    include 'graphics/gfx.i'
    ENDC

    IFND    GRAPHICS_GFXNODES_I
    include 'graphics/gfxnodes.i'
    ENDC

    IFND    GRAPHICS_VIEW_I
    include 'graphics/view.i'
    ENDC

    IFND    EXEC_SEMAPHORES_I
    include 'exec/semaphores.i'
    ENDC

    STRUCTURE   AnalogSignalInterval,0
    UWORD       asi_Start
    UWORD       asi_Stop
    LABEL       asi_SIZEOF

    STRUCTURE   SpecialMonitor,XLN_SIZE
    UWORD       spm_Flags
    APTR        spm_do_monitor
    APTR        spm_reserved1
    APTR        spm_reserved2
    APTR        spm_reserved3
    STRUCT      spm_hblank,asi_SIZEOF
    STRUCT      spm_vblank,asi_SIZEOF
    STRUCT      spm_hsync,asi_SIZEOF
    STRUCT      spm_vsync,asi_SIZEOF
    LABEL       spm_SIZEOF

    STRUCTURE   MonitorSpec,XLN_SIZE
    UWORD       ms_Flags
    LONG        ms_ratioh
    LONG        ms_ratiov
    UWORD       ms_total_rows
    UWORD       ms_total_colorclocks
    UWORD       ms_DeniseMaxDisplayColumn
    UWORD       ms_BeamCon0
    UWORD       ms_min_row
    APTR        ms_Special
    UWORD       ms_OpenCount
    APTR        ms_transform
    APTR        ms_translate
    APTR        ms_scale
    UWORD       ms_xoffset
    UWORD       ms_yoffset
    STRUCT      ms_LegalView,ra_SIZEOF
    APTR        ms_maxoscan
    APTR        ms_videoscan
    UWORD       ms_DeniseMinDisplayColumn
    UWORD       ms_DisplayCompatible
    STRUCT      ms_DisplayInfoDataBase,LH_SIZE
    STRUCT      ms_DIDBSemaphore,SS_SIZE
    ULONG       ms_reserved00
    ULONG       ms_reserved01
    LABEL       ms_SIZEOF


    BITDEF      MS,REQUEST_NTSC,0
    BITDEF      MS,REQUEST_PAL,1
    BITDEF      MS,REQUEST_SPECIAL,2
    BITDEF      MS,REQUEST_A2024,3

STANDARD_VIEW_X equ $81
STANDARD_VIEW_Y equ $2C

    ENDC        ; GRAPHICS_MONITOR_I
```

```
        IFND    GRAPHICS_RASTPORT_I
GRAPHICS_RASTPORT_I     SET     1
**
**      $Filename: graphics/rastport.i $
**      $Release: 2.04 $
**      $Revision: 37.0 $
**      $Date: 91/01/07 $
**
**
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

    IFND    EXEC_TYPES_I
    include 'exec/types.i'
    ENDC

    IFND    GRAPHICS_GFX_I
      include "graphics/gfx.i"
    ENDC

*------ TR : TmpRas --------------------------------------------------

 STRUCTURE  TmpRas,0
   APTR     tr_RasPtr           ; *WORD
   LONG     tr_Size
   LABEL    tr_SIZEOF

*------ GelsInfo

   STRUCTURE   GelsInfo,0
   BYTE     gi_sprRsrvd         * flag of which sprites to reserve from
*                               * vsprite system
   BYTE     gi_Flags            * reserved for system use
   APTR     gi_gelHead
   APTR     gi_gelTail          * dummy vSprites for list management
* pointer to array of 8 WORDS for sprite available lines
   APTR     gi_nextLine
* pointer to array of 8 pointers for color-last-assigned to vSprites
   APTR     gi_lastColor
   APTR     gi_collHandler      * addresses of collision routines
   WORD    gi_leftmost
   WORD    gi_rightmost
   WORD    gi_topmost
   WORD    gi_bottommost
   APTR     gi_firstBlissObj
   APTR     gi_lastBlissObj     * system use only
   LABEL    gi_SIZEOF

*------ RP_Flags ------
   BITDEF   RP,FRST_DOT,0       ; draw the first dot of this line ?
   BITDEF   RP,ONE_DOT,1        ; use one dot mode for drawing lines
   BITDEF   RP,DBUFFER,2        ; flag set when RastPorts are double-buffered
*                               ;   (only used for bobs)
   BITDEF   RP,AREAOUTLINE,3    ; used by areafiller
   BITDEF   RP,NOCROSSFILL,5    ; used by areafiller

*------ RP_DrawMode ------
RP_JAM1         EQU 0
RP_JAM2         EQU 1
RP_COMPLEMENT   EQU 2
RP_INVERSVID    EQU 4           ; inverse video for drawing modes

*------ RP_TxFlags ------
   BITDEF   RP,TXSCALE,0
```

graphics/rastport.i

Page 2

```

 STRUCTURE  RastPort,0
   LONG     rp_Layer
   LONG     rp_BitMap
   LONG     rp_AreaPtrn
   LONG     rp_TmpRas
   LONG     rp_AreaInfo
   LONG     rp_GelsInfo
   BYTE     rp_Mask
   BYTE     rp_FgPen
   BYTE     rp_BgPen
   BYTE     rp_AOLPen
   BYTE     rp_DrawMode
   BYTE     rp_AreaPtSz
   BYTE     rp_linpatcnt
   BYTE     rp_Dummy
   WORD     rp_Flags
   WORD     rp_LinePtrn
   WORD     rp_cp_x
   WORD     rp_cp_y
   STRUCT   rp_minterms,8
   WORD     rp_PenWidth
   WORD     rp_PenHeight
   LONG     rp_Font
   BYTE     rp_AlgoStyle
   BYTE     rp_TxFlags
   WORD     rp_TxHeight
   WORD     rp_TxWidth
   WORD     rp_TxBaseline
   WORD     rp_TxSpacing
   APTR     rp_RP_User
   STRUCT   rp_longreserved,8
        ifnd    GFX_RASTPORT_1_2
   STRUCT   rp_wordreserved,14
   STRUCT   rp_reserved,8
        endc
   LABEL    rp_SIZEOF

 STRUCTURE  AreaInfo,0
   LONG     ai_VctrTbl
   LONG     ai_VctrPtr
   LONG     ai_FlagTbl
   LONG     ai_FlagPtr
   WORD     ai_Count
   WORD     ai_MaxCount
   WORD     ai_FirstX
   WORD     ai_FirstY
   LABEL    ai_SIZEOF

ONE_DOTn    equ     1
ONE_DOT     equ     $2      * 1<<ONE_DOTn
FRST_DOTn   equ     0
FRST_DOT    equ     1       * 1<<FRST_DOTn

        ENDC    ; GRAPHICS_RASTPORT_I
```

graphics/regions.i

Page 1

```
        IFND    GRAPHICS_REGIONS_I
GRAPHICS_REGIONS_I      SET     1
**
**      $Filename: graphics/regions.i $
**      $Release: 2.04 $
**      $Revision: 37.0 $
**      $Date: 91/01/07 $
**
**
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

    IFND    EXEC_TYPES_I
    include 'exec/types.i'
    ENDC

   IFND  GRAPHICS_GFX_I
   include  "graphics/gfx.i"
   ENDC

    STRUCTURE   Region,0
      STRUCT   rg_bounds,ra_SIZEOF
      APTR  rg_RegionRectangle
   LABEL    rg_SIZEOF

   STRUCTURE   RegionRectangle,0
      APTR  rr_Next
      APTR  rr_Prev
      STRUCT   rr_bounds,ra_SIZEOF
   LABEL    rr_SIZEOF

        ENDC    ; GRAPHICS_REGIONS_I
```

graphics/scale.i

```
        IFND    GRAPHICS_SCALE_I
GRAPHICS_SCALE_I        SET     1
**
**      $Filename: graphics/scale.i $
**      $Release: 2.04 $
**      $Revision: 37.0 $
**      $Date: 91/01/07 $
**
**      structure argument to BitMapScale()
**
**      (C) Copyright 1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

    IFND    EXEC_TYPES_I
    include 'exec/types.i'
    ENDC

 STRUCTURE      BitScaleArgs,0
    UWORD   bsa_SrcX            ; source origin
    UWORD   bsa_SrcY
    UWORD   bsa_SrcWidth        ; source size
    UWORD   bsa_SrcHeight
    UWORD   bsa_XSrcFactor      ; scale factor denominators
    UWORD   bsa_YSrcFactor
    UWORD   bsa_DestX           ; destination origin
    UWORD   bsa_DestY
    UWORD   bsa_DestWidth       ; destination size result
    UWORD   bsa_DestHeight
    UWORD   bsa_XDestFactor     ; scale factor numerators
    UWORD   bsa_YDestFactor
    APTR    bsa_SrcBitMap       ; source BitMap
    APTR    bsa_DestBitMap      ; destination BitMap
    ULONG   bsa_Flags           ; reserved.  Must be zero!
    UWORD   bsa_XDDA            ; reserved
    UWORD   bsa_YDDA
    LONG    bsa_Reserved1
    LONG    bsa_Reserved2
    LABEL   bsa_SIZEOF

        ENDC    ; GRAPHICS_SCALE_I
```

graphics/sprite.i

```
        IFND    GRAPHICS_SPRITE_I
GRAPHICS_SPRITE_I       SET     1
**
**      $Filename: graphics/sprite.i $
**      $Release: 2.04 $
**      $Revision: 37.0 $
**      $Date: 91/01/07 $
**
**
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

    IFND    EXEC_TYPES_I
    include 'exec/types.i'
    ENDC

   STRUCTURE   SimpleSprite,0
   APTR        ss_posctldata
   WORD        ss_height
   WORD        ss_x
   WORD        ss_y
   WORD        ss_num
   LABEL       ss_SIZEOF

        ENDC    ; GRAPHICS_SPRITE_I
```

graphics/text.i

```
        IFND    GRAPHICS_TEXT_I
GRAPHICS_TEXT_I SET     1
**
**      $Filename: graphics/text.i $
**      $Release: 2.04 $
**      $Revision: 37.0 $
**      $Date: 91/01/07 $
**
**      graphics library text structures
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

        IFND    EXEC_PORTS_I
        INCLUDE "exec/ports.i"
        ENDC    ; EXEC_PORTS_I

        IFND    UTILITY_TAGITEM_I
        INCLUDE "utility/tagitem.i"
        ENDC    ; UTILITY_TAGITEM_I

*------ Font Styles ------------------------------------------------
FS_NORMAL       EQU     0           ; normal text (no style attributes set)
    BITDEF  FS,UNDERLINED,0         ; underlined (under baseline)
    BITDEF  FS,BOLD,1               ; bold face text (ORed w/ shifted right 1)
    BITDEF  FS,ITALIC,2             ; italic (slanted 1:2 right)
    BITDEF  FS,EXTENDED,3           ; extended face (must be designed)

    BITDEF  FS,COLORFONT,6          ; this uses ColorTextFont structure
    BITDEF  FS,TAGGED,7             ; the TextAttr is really an TTextAttr,

*------ Font Flags -------------------------------------------------
    BITDEF  FP,ROMFONT,0            ; font is in rom
    BITDEF  FP,DISKFONT,1           ; font is from diskfont.library
    BITDEF  FP,REVPATH,2            ; designed path is reversed (e.g. left)
    BITDEF  FP,TALLDOT,3            ; designed for hires non-interlaced
    BITDEF  FP,WIDEDOT,4            ; designed for lores interlaced
    BITDEF  FP,PROPORTIONAL,5       ; character sizes vary from tf_XSize
    BITDEF  FP,DESIGNED,6           ; size is "designed", not constructed
*                                   ; note: if you do not set this bit in your
*                                   ; textattr, then a font may be constructed
*                                   ; for you by scaling an existing rom or disk
*                                   ; font (under V36 and above).
    ;-- bit 7 is always clear for fonts on the graphics font list
    BITDEF  FP,REMOVED,7            ; the font has been removed


******* TextAttr node ***************************************************
 STRUCTURE  TextAttr,0
    APTR    ta_Name                 ; name of the desired font
    UWORD   ta_YSize                ; height of the desired font
    UBYTE   ta_Style                ; desired font style
    UBYTE   ta_Flags                ; font preferences flags
    LABEL   ta_SIZEOF

 STRUCTURE  TTextAttr,0
    APTR    tta_Name                ; name of the desired font
    UWORD   tta_YSize               ; height of the desired font
    UBYTE   tta_Style               ; desired font style
    UBYTE   tta_Flags               ; font preferences flags
    APTR    tta_Tags                ; extended attributes
    LABEL   tta_SIZEOF

******* Text Tags *******************************************************
TA_DeviceDPI    EQU     1!TAG_USER      ; Tag value is Point union:
```

graphics/text.i

```
                                        ; Hi word XDPI, Lo word YDPI


MAXFONTMATCHWEIGHT      EQU     32767   ; perfect match from WeighTAMatch


******* TextFont node ***************************************************
 STRUCTURE      TextFont,MN_SIZE ; reply message for font removal
                                ; font name in LN_NAME  \ used in this
    UWORD   tf_YSize            ; font height           | order to best
    UBYTE   tf_Style            ; font style            | match a font
    UBYTE   tf_Flags            ; preference attributes / request.
    UWORD   tf_XSize            ; nominal font width
    UWORD   tf_Baseline         ; distance from the top of char to baseline
    UWORD   tf_BoldSmear        ; smear to affect a bold enhancement

    UWORD   tf_Accessors        ; access count

    UBYTE   tf_LoChar           ; the first character described here
    UBYTE   tf_HiChar           ; the last character described here
    APTR    tf_CharData         ; the bit character data

    UWORD   tf_Modulo           ; the row modulo for the strike font data
    APTR    tf_CharLoc          ; ptr to location data for the strike font
                                ;   2 words: bit offset then size
    APTR    tf_CharSpace        ; ptr to words of proportional spacing data
    APTR    tf_CharKern         ; ptr to words of kerning data
    LABEL   tf_SIZEOF

tf_Extension    EQU     MN_REPLYPORT

;------ tfe_Flags0 (partial definition) ------------------------------
    BITDEF  TE0,NOREMFONT,0     ; disallow RemFont for this font

 STRUCTURE      TextFontExtension,0     ; this structure is read-only
    UWORD   tfe_MatchWord       ; a magic cookie for the extension
    UBYTE   tfe_Flags0          ; (system private flags)
    UBYTE   tfe_Flags1          ; (system private flags)
    APTR    tfe_BackPtr         ; validation of compilation
    APTR    tfe_OrigReplyPort   ; original value in tf_Extension
    APTR    tfe_Tags            ; Text Tags for the font
    APTR    tfe_OFontPatchS     ; (system private use)
    APTR    tfe_OFontPatchK     ; (system private use)
    ; this space is reserved for future expansion
    LABEL   tfe_SIZEOF          ; (but allocated only by the system)


******* ColorTextFont node *********************************************
;------ ctf_Flags ------------------------------------------------------
CT_COLORFONT    EQU     $0001   ; color map contains designer's colors
CT_GREYFONT     EQU     $0002   ; color map describes even-stepped brightnesses
                                ; from low to high
CT_ANTIALIAS    EQU     $0004   ; zero background thru fully saturated char

        BITDEF  CT,MAPCOLOR,0   ; map ctf_FgColor to the rp_FgPen if the former
                                ; is a valid color within ctf_Low..ctf_High

;------ ColorFontColors ----------------------------------------------
 STRUCTURE      ColorFontColors,0
    UWORD   cfc_Reserved        ; must be zero
    UWORD   cfc_Count           ; number of entries in cfc_ColorTable
    APTR    cfc_ColorTable      ; 4 bit per component color map packed xRGB
    LABEL   cfc_SIZEOF

;------ ColorTextFont ------------------------------------------------
 STRUCTURE      ColorTextFont,tf_SIZEOF
```

graphics/text.i

```
    UWORD   ctf_Flags           ; extended flags
    UBYTE   ctf_Depth           ; number of bit planes
    UBYTE   ctf_FgColor         ; color that is remapped to FgPen
    UBYTE   ctf_Low             ; lowest color represented here
    UBYTE   ctf_High            ; highest color represented here
    UBYTE   ctf_PlanePick       ; PlanePick ala Images
    UBYTE   ctf_PlaneOnOff      ; PlaneOnOff ala Images
    APTR    ctf_ColorFontColors ; struct ColorFontColors * for font
    STRUCT  ctf_CharData,8*4    ; pointers to bit planes ala tf_CharData
    LABEL   ctf_SIZEOF

******* TextExtent node *********************************************
 STRUCTURE      TextExtent,0
    UWORD   te_Width            ; same as TextLength
    UWORD   te_Height           ; same as tf_YSize
    STRUCT  te_Extent,8         ; WORD MinX, MinY, MaxX, MaxY relative to CP
    LABEL   te_SIZEOF

        ENDC    ; GRAPHICS_TEXT_I
```

graphics/videocontrol.i

```
    IFND    GRAPHICS_VIDEOCONTROL_I
GRAPHICS_VIDEOCONTROL_I SET 1
**
**      $Filename: graphics/videocontrol.i $
**      $Release: 2.04 $
**      $Revision: 37.0 $
**      $Date: 91/01/07 $
**
**      graphics videocontrol definitions
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

    IFND    EXEC_TYPES_I
    include 'exec/types.i'
    ENDC

    IFND    UTILITY_TAGITEM_I
    include 'utility/tagitem.i'
    ENDC


VTAG_END_CM                 equ     $00000000
VTAG_CHROMAKEY_CLR          equ     $80000000
VTAG_CHROMAKEY_SET          equ     $80000001
VTAG_BITPLANEKEY_CLR        equ     $80000002
VTAG_BITPLANEKEY_SET        equ     $80000003
VTAG_BORDERBLANK_CLR        equ     $80000004
VTAG_BORDERBLANK_SET        equ     $80000005
VTAG_BORDERNOTRANS_CLR      equ     $80000006
VTAG_BORDERNOTRANS_SET      equ     $80000007
VTAG_CHROMA_PEN_CLR         equ     $80000008
VTAG_CHROMA_PEN_SET         equ     $80000009
VTAG_CHROMA_PLANE_SET       equ     $8000000A
VTAG_ATTACH_CM_SET          equ     $8000000B
VTAG_NEXTBUF_CM             equ     $8000000C
VTAG_BATCH_CM_CLR           equ     $8000000D
VTAG_BATCH_CM_SET           equ     $8000000E
VTAG_NORMAL_DISP_GET        equ     $8000000F
VTAG_NORMAL_DISP_SET        equ     $80000010
VTAG_COERCE_DISP_GET        equ     $80000011
VTAG_COERCE_DISP_SET        equ     $80000012
VTAG_VIEWPORTEXTRA_GET      equ     $80000013
VTAG_VIEWPORTEXTRA_SET      equ     $80000014
VTAG_CHROMAKEY_GET          equ     $80000015
VTAG_BITPLANEKEY_GET        equ     $80000016
VTAG_BORDERBLANK_GET        equ     $80000017
VTAG_BORDERNOTRANS_GET      equ     $80000018
VTAG_CHROMA_PEN_GET         equ     $80000019
VTAG_CHROMA_PLANE_GET       equ     $8000001A
VTAG_ATTACH_CM_GET          equ     $8000001B
VTAG_BATCH_CM_GET           equ     $8000001C
VTAG_BATCH_ITEMS_GET        equ     $8000001D
VTAG_BATCH_ITEMS_SET        equ     $8000001E
VTAG_BATCH_ITEMS_ADD        equ     $8000001F
VTAG_VPMODEID_GET           equ     $80000020
VTAG_VPMODEID_SET           equ     $80000021
VTAG_VPMODEID_CLR           equ     $80000022
VTAG_USERCLIP_GET           equ     $80000023
VTAG_USERCLIP_SET           equ     $80000024
VTAG_USERCLIP_CLR           equ     $80000025


    ENDC        ; GRAPHICS_VIDEOCONTROL_I
    IFND    GRAPHICS_VIEW_I
GRAPHICS_VIEW_I SET 1
**
**      $Filename: graphics/view.i $
**      $Release: 2.04 $
**      $Revision: 37.0 $
**      $Date: 91/01/07 $
**
**      graphics view/viewport definitions
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

    IFND    EXEC_TYPES_I
    include 'exec/types.i'
    ENDC

    IFND    GRAPHICS_GFX_I
    include 'graphics/gfx.i'
    ENDC

    IFND    GRAPHICS_COPPER_I
    include 'graphics/copper.i'
    ENDC

    IFND    GRAPHICS_GFXNODES_I
    include 'graphics/gfxnodes.i'
    ENDC

GENLOCK_VIDEO           EQU     $2
V_LACE                  EQU     $4
V_SUPERHIRES            EQU     $20
V_PFBA                  EQU     $40
V_EXTRA_HALFBRITE       EQU     $80
GENLOCK_AUDIO           EQU     $100
V_DUALPF                EQU     $400
V_HAM                   EQU     $800
V_EXTENDED_MODE EQU     $1000
V_VP_HIDE               EQU     $2000
V_SPRITES               EQU     $4000
V_HIRES         EQU     $8000

EXTEND_VSTRUCT          EQU     $1000

VPF_DENISE      EQU     $80
VPF_A2024       EQU     $40
VPF_AGNUS       EQU     $20
VPF_TENHZ       EQU     $20
VPF_ILACE       EQU     $10

   STRUCTURE   ColorMap,0
        BYTE    cm_Flags
        BYTE    cm_Type
        WORD    cm_Count
        APTR    cm_ColorTable
        APTR    cm_vpe
        APTR    cm_TransparencyBits
        BYTE    cm_TransparenyPlane
        BYTE    cm_reserved1
        WORD    cm_reserved2
        APTR    cm_vp
        APTR    cm_NormalDisplayInfo
        APTR    cm_CoerceDisplayInfo
        APTR    cm_batch_items
        LONG    cm_VPModeID
```

```
   LABEL cm_SIZEOF

COLORMAP_TYPE_V1_2      EQU     $00
COLORMAP_TYPE_V1_4      EQU     $01
COLORMAP_TYPE_V36 EQU COLORMAP_TYPE_V1_4        ; use this definition

COLORMAP_TRANSPARENCY   EQU     $01
COLORPLANE_TRANSPARENCY EQU     $02
BORDER_BLANKING         EQU     $04
BORDER_NOTRANSPARENCY   EQU     $08
VIDEOCONTROL_BATCH      EQU     $10
USER_COPPER_CLIP        EQU     $20

   STRUCTURE     ViewPort,0
   LONG    vp_Next
   LONG    vp_ColorMap
   LONG    vp_DspIns
   LONG    vp_SprIns
   LONG    vp_ClrIns
   LONG    vp_UCopIns
   WORD    vp_DWidth
   WORD    vp_DHeight
   WORD    vp_DxOffset
   WORD    vp_DyOffset
   WORD    vp_Modes
   BYTE    vp_SpritePriorities
   BYTE    vp_ExtendedModes
   APTR    vp_RasInfo
   LABEL   vp_SIZEOF


   STRUCTURE View,0
   LONG    v_ViewPort
   LONG    v_LOFCprList
   LONG    v_SHFCprList
   WORD    v_DyOffset
   WORD    v_DxOffset
   WORD    v_Modes
   LABEL   v_SIZEOF


   STRUCTURE ViewExtra,XLN_SIZE
   APTR    ve_View
   APTR    ve_Monitor
   LABEL   ve_SIZEOF


   STRUCTURE ViewPortExtra,XLN_SIZE
   APTR    vpe_ViewPort
   STRUCT  vpe_DisplayClip,ra_SIZEOF
   LABEL   vpe_SIZEOF


   STRUCTURE  collTable,0
   LONG    cp_collPtrs,16
   LABEL   cp_SIZEOF


   STRUCTURE  RasInfo,0
   APTR    ri_Next
   LONG    ri_BitMap
   WORD    ri_RxOffset
   WORD    ri_RyOffset
   LABEL   ri_SIZEOF

        ENDC    ; GRAPHICS_VIEW_I
```

hardware/adkbits.i

```
        IFND    HARDWARE_ADKBITS_I
HARDWARE_ADKBITS_I      SET     1
**
**      $Filename: hardware/adkbits.i $
**      $Release: 2.04 $
**      $Revision: 36.2 $
**      $Date: 90/07/10 $
**
**      bit definitions for adkcon register
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

ADKB_SETCLR     EQU   15 ; standard set/clear bit
ADKB_PRECOMP1   EQU   14 ; two bits of precompensation
ADKB_PRECOMP0   EQU   13
ADKB_MFMPREC    EQU   12 ; use mfm style precompensation
ADKB_UARTBRK    EQU   11 ; force uart output to zero
ADKB_WORDSYNC   EQU   10 ; enable DSKSYNC register matching
ADKB_MSBSYNC    EQU   9  ; (Apple GCR Only) sync on MSB for reading
ADKB_FAST       EQU   8  ; 1 -> 2 us/bit (mfm), 2 -> 4 us/bit (gcr)
ADKB_USE3PN     EQU   7  ; use aud chan 3 to modulate period of ??
ADKB_USE2P3     EQU   6  ; use aud chan 2 to modulate period of 3
ADKB_USE1P2     EQU   5  ; use aud chan 1 to modulate period of 2
ADKB_USE0P1     EQU   4  ; use aud chan 0 to modulate period of 1
ADKB_USE3VN     EQU   3  ; use aud chan 3 to modulate volume of ??
ADKB_USE2V3     EQU   2  ; use aud chan 2 to modulate volume of 3
ADKB_USE1V2     EQU   1  ; use aud chan 1 to modulate volume of 2
ADKB_USE0V1     EQU   0  ; use aud chan 0 to modulate volume of 1

ADKF_SETCLR     EQU   (1<<15)
ADKF_PRECOMP1   EQU   (1<<14)
ADKF_PRECOMP0   EQU   (1<<13)
ADKF_MFMPREC    EQU   (1<<12)
ADKF_UARTBRK    EQU   (1<<11)
ADKF_WORDSYNC   EQU   (1<<10)
ADKF_MSBSYNC    EQU   (1<<9)
ADKF_FAST       EQU   (1<<8)
ADKF_USE3PN     EQU   (1<<7)
ADKF_USE2P3     EQU   (1<<6)
ADKF_USE1P2     EQU   (1<<5)
ADKF_USE0P1     EQU   (1<<4)
ADKF_USE3VN     EQU   (1<<3)
ADKF_USE2V3     EQU   (1<<2)
ADKF_USE1V2     EQU   (1<<1)
ADKF_USE0V1     EQU   (1<<0)

ADKF_PRE000NS   EQU   0                         ; 000 ns of precomp
ADKF_PRE140NS   EQU   (ADKF_PRECOMP0)           ; 140 ns of precomp
ADKF_PRE280NS   EQU   (ADKF_PRECOMP1)           ; 280 ns of precomp
ADKF_PRE560NS   EQU   (ADKF_PRECOMP0!ADKF_PRECOMP1) ; 560 ns of precomp

        ENDC    ; HARDWARE_ADKBITS_I
```

hardware/blit.i

```
        IFND    HARDWARE_BLIT_I
HARDWARE_BLIT_I SET     1
**
**      $Filename: hardware/blit.i $
**      $Release: 2.04 $
**      $Revision: 36.3 $
**      $Date: 90/11/05 $
**
**      Defines for direct hardware use of the blitter.
**
**      (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**

        IFND    EXEC_TYPES_I
        INCLUDE "exec/types.i"
        ENDC    ;EXEC_TYPES_I



    STRUCTURE bltnode,0
    LONG  bn_n
    LONG  bn_function
    BYTE  bn_stat
    BYTE  bn_dummy
    WORD  bn_blitsize
    WORD  bn_beamsync
    LONG  bn_cleanup
    LABEL bn_SIZEOF

* bit defines used by blit queuer
CLEANMEn      equ 6
CLEANME       equ 1<<CLEANMEn

* include file for blitter */
HSIZEBITS     equ   6
VSIZEBITS     equ   16-HSIZEBITS
HSIZEMASK     equ   $3f          /* 2^6 -- 1 */
VSIZEMASK     equ   $3FF         /* 2^10 - 1 */

* all agnii support horizontal blit of at least 1024 bits (128 bytes) wide */
* some agnii support horizontal blit of up to 32768 bits (4096 bytes) wide */

        IFD     NO_BIG_BLITS
MAXBYTESPERROW EQU   128
        ENDC

        IFND    NO_BIG_BLITS
MINBYTESPERROW EQU   128
MAXBYTESPERROW EQU   4096
        ENDC

* definitions for blitter control register 0 */

ABC           equ   $80
ABNC          equ   $40
ANBC          equ   $20
ANBNC         equ   $10
NABC          equ   $8
NABNC         equ   $4
NANBC         equ   $2
NANBNC        equ   $1

BCOB_DEST     equ     8
BCOB_SRCC     equ     9
BCOB_SRCB     equ     10
```

hardware/blit.i

Page 2

```
BCOB_SRCA   equ     11
BCOF_DEST   equ   $100
BCOF_SRCC   equ   $200
BCOF_SRCB   equ   $400
BCOF_SRCA   equ   $800

BC1F_DESC   equ 2

DEST        equ   $100
SRCC        equ   $200
SRCB        equ   $400
SRCA        equ   $800

ASHIFTSHIFT equ   12 /* bits to right align ashift value */
BSHIFTSHIFT equ   12 /* bits to right align bshift value */

* definations for blitter control register 1 */
LINEMODE    equ   $1
FILL_OR     equ   $8
FILL_XOR    equ   $10
FILL_CARRYIN   equ   $4
ONEDOT      equ   $2
OVFLAG      equ   $20
SIGNFLAG    equ   $40
BLITREVERSE equ   $2

SUD         equ   $10
SUL         equ   $8
AUL         equ   $4

OCTANT8     equ   24
OCTANT7     equ   4
OCTANT6     equ   12
OCTANT5     equ   28
OCTANT4     equ   20
OCTANT3     equ   8
OCTANT2     equ   0
OCTANT1     equ   16

        ENDC    ; HARDWARE_BLIT_I
```

hardware/cia.i

Page 1

```
        IFND    HARDWARE_CIA_I
HARDWARE_CIA_I  SET     1
**
**      $Filename: hardware/cia.i $
**      $Release: 2.04 $
**      $Revision: 36.2 $
**      $Date: 90/07/10 $
**
**      registers and bits in the Complex Interface Adapter (CIA) chip
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

*
* _ciaa is on an ODD address (e.g. the low byte) -- $bfe001
* _ciab is on an EVEN address (e.g. the high byte) -- $bfd000
*
* do this to get the definitions:
*    XREF _ciaa
*    XREF _ciab
*


* cia register offsets
ciapra           EQU   $0000
ciaprb           EQU   $0100
ciaddra   EQU   $0200
ciaddrb   EQU   $0300
ciatalo   EQU   $0400
ciatahi   EQU   $0500
ciatblo   EQU   $0600
ciatbhi   EQU   $0700
ciatodlow        EQU   $0800
ciatodmid        EQU   $0900
ciatodhi         EQU   $0A00
ciasdr           EQU   $0C00
ciaicr           EQU   $0D00
ciacra           EQU   $0E00
ciacrb           EQU   $0F00

* interrupt control register bit numbers
CIAICRB_TA       EQU   0
CIAICRB_TB       EQU   1
CIAICRB_ALRM     EQU   2
CIAICRB_SP       EQU   3
CIAICRB_FLG      EQU   4
CIAICRB_IR       EQU   7
CIAICRB_SETCLR   EQU   7

* control register A bit numbers
CIACRAB_START    EQU   0
CIACRAB_PBON     EQU   1
CIACRAB_OUTMODE  EQU   2
CIACRAB_RUNMODE  EQU   3
CIACRAB_LOAD     EQU   4
CIACRAB_INMODE   EQU   5
CIACRAB_SPMODE   EQU   6
CIACRAB_TODIN    EQU   7

* control register B bit numbers
CIACRBB_START    EQU   0
CIACRBB_PBON     EQU   1
CIACRBB_OUTMODE  EQU   2
CIACRBB_RUNMODE  EQU   3
CIACRBB_LOAD     EQU   4
```

```
CIACRBB_INMODE0   EQU   5
CIACRBB_INMODE1   EQU   6
CIACRBB_ALARM     EQU   7

* interrupt control register bit masks
CIAICRF_TA        EQU   (1<<0)
CIAICRF_TB        EQU   (1<<1)
CIAICRF_ALRM      EQU   (1<<2)
CIAICRF_SP        EQU   (1<<3)
CIAICRF_FLG       EQU   (1<<4)
CIAICRF_IR        EQU   (1<<7)
CIAICRF_SETCLR    EQU   (1<<7)

* control register A bit masks
CIACRAF_START     EQU   (1<<0)
CIACRAF_PBON      EQU   (1<<1)
CIACRAF_OUTMODE   EQU   (1<<2)
CIACRAF_RUNMODE   EQU   (1<<3)
CIACRAF_LOAD      EQU   (1<<4)
CIACRAF_INMODE    EQU   (1<<5)
CIACRAF_SPMODE    EQU   (1<<6)
CIACRAF_TODIN     EQU   (1<<7)

* control register B bit masks
CIACRBF_START     EQU   (1<<0)
CIACRBF_PBON      EQU   (1<<1)
CIACRBF_OUTMODE   EQU   (1<<2)
CIACRBF_RUNMODE   EQU   (1<<3)
CIACRBF_LOAD      EQU   (1<<4)
CIACRBF_INMODE0   EQU   (1<<5)
CIACRBF_INMODE1   EQU   (1<<6)
CIACRBF_ALARM     EQU   (1<<7)

* control register B INMODE masks
CIACRBF_IN_PHI2   EQU   0
CIACRBF_IN_CNT    EQU   (CIACRBF_INMODE0)
CIACRBF_IN_TA     EQU   (CIACRBF_INMODE1)
CIACRBF_IN_CNT_TA EQU   (CIACRBF_INMODE0!CIACRBF_INMODE1)


*
* Port definitions -- what each bit in a cia peripheral register is tied to
*

* ciaa port A (0xbfe001)
CIAB_GAMEPORT1    EQU   (7)   * gameport 1, pin 6 (fire button*)
CIAB_GAMEPORT0    EQU   (6)   * gameport 0, pin 6 (fire button*)
CIAB_DSKRDY       EQU   (5)   * disk ready*
CIAB_DSKTRACK0    EQU   (4)   * disk on track 00*
CIAB_DSKPROT      EQU   (3)   * disk write protect*
CIAB_DSKCHANGE    EQU   (2)   * disk change*
CIAB_LED          EQU   (1)   * led light control (0==>bright)
CIAB_OVERLAY      EQU   (0)   * memory overlay bit

* ciaa port B (0xbfe101) -- parallel port

* ciab port A (0xbfd000) -- serial and printer control
CIAB_COMDTR       EQU   (7)   * serial Data Terminal Ready*
CIAB_COMRTS       EQU   (6)   * serial Request to Send*
CIAB_COMCD        EQU   (5)   * serial Carrier Detect*
CIAB_COMCTS       EQU   (4)   * serial Clear to Send*
CIAB_COMDSR       EQU   (3)   * serial Data Set Ready*
CIAB_PRTRSEL      EQU   (2)   * printer SELECT
CIAB_PRTRPOUT     EQU   (1)   * printer paper out
CIAB_PRTRBUSY     EQU   (0)   * printer busy

```

```
* ciab port B (0xbfd100) -- disk control
CIAB_DSKMOTOR     EQU   (7)   * disk motorr*
CIAB_DSKSEL3      EQU   (6)   * disk select unit 3*
CIAB_DSKSEL2      EQU   (5)   * disk select unit 2*
CIAB_DSKSEL1      EQU   (4)   * disk select unit 1*
CIAB_DSKSEL0      EQU   (3)   * disk select unit 0*
CIAB_DSKSIDE      EQU   (2)   * disk side select*
CIAB_DSKDIREC     EQU   (1)   * disk direction of seek*
CIAB_DSKSTEP      EQU   (0)   * disk step heads*

* ciaa port A (0xbfe001)
CIAF_GAMEPORT1    EQU   (1<<7)
CIAF_GAMEPORT0    EQU   (1<<6)
CIAF_DSKRDY       EQU   (1<<5)
CIAF_DSKTRACK0    EQU   (1<<4)
CIAF_DSKPROT      EQU   (1<<3)
CIAF_DSKCHANGE    EQU   (1<<2)
CIAF_LED          EQU   (1<<1)
CIAF_OVERLAY      EQU   (1<<0)

* ciaa port B (0xbfe101) -- parallel port

* ciab port A (0xbfd000) -- serial and printer control
CIAF_COMDTR       EQU   (1<<7)
CIAF_COMRTS       EQU   (1<<6)
CIAF_COMCD        EQU   (1<<5)
CIAF_COMCTS       EQU   (1<<4)
CIAF_COMDSR       EQU   (1<<3)
CIAF_PRTRSEL      EQU   (1<<2)
CIAF_PRTRPOUT     EQU   (1<<1)
CIAF_PRTRBUSY     EQU   (1<<0)

* ciab port B (0xbfd100) -- disk control
CIAF_DSKMOTOR     EQU   (1<<7)
CIAF_DSKSEL3      EQU   (1<<6)
CIAF_DSKSEL2      EQU   (1<<5)
CIAF_DSKSEL1      EQU   (1<<4)
CIAF_DSKSEL0      EQU   (1<<3)
CIAF_DSKSIDE      EQU   (1<<2)
CIAF_DSKDIREC     EQU   (1<<1)
CIAF_DSKSTEP      EQU   (1<<0)

        ENDC    ; HARDWARE_CIA_I
```

hardware/custom.i Page 1

```
        IFND    HARDWARE_CUSTOM_I
HARDWARE_CUSTOM_I       SET     1
**
**      $Filename: hardware/custom.i $
**      $Release: 2.04 $
**      $Revision: 36.5 $
**      $Date: 90/11/05 $
**
**      Offsets of Amiga custom chip registers
**
**      (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**

*
* do this to get base of custom registers:
*  XREF _custom;
*

bltddat     EQU   $000
dmaconr     EQU   $002
vposr       EQU   $004
vhposr      EQU   $006
dskdatr     EQU   $008
joy0dat     EQU   $00A
joy1dat     EQU   $00C
clxdat      EQU   $00E

adkconr     EQU   $010
pot0dat     EQU   $012
pot1dat     EQU   $014
potinp      EQU   $016
serdatr     EQU   $018
dskbytr     EQU   $01A
intenar     EQU   $01C
intreqr     EQU   $01E

dskpt       EQU   $020
dsklen      EQU   $024
dskdat      EQU   $026
refptr      EQU   $028
vposw       EQU   $02A
vhposw      EQU   $02C
copcon      EQU   $02E
serdat      EQU   $030
serper      EQU   $032
potgo       EQU   $034
joytest     EQU   $036
strequ      EQU   $038
strvbl      EQU   $03A
strhor      EQU   $03C
strlong     EQU   $03E

bltcon0     EQU   $040
bltcon1     EQU   $042
bltafwm     EQU   $044
bltalwm     EQU   $046
bltcpt      EQU   $048
bltbpt      EQU   $04C
bltapt      EQU   $050
bltdpt      EQU   $054
bltsize     EQU   $058
bltcon01    EQU   $05B          ; note: byte access only
bltsizv     EQU   $05C
bltsizh     EQU   $05E

```

hardware/custom.i Page 2

```
bltcmod     EQU   $060
bltbmod     EQU   $062
bltamod     EQU   $064
bltdmod     EQU   $066

bltcdat     EQU   $070
bltbdat     EQU   $072
bltadat     EQU   $074

deniseid    EQU   $07C
dsksync     EQU   $07E

cop1lc      EQU   $080
cop2lc      EQU   $084
copjmp1     EQU   $088
copjmp2     EQU   $08A
copins      EQU   $08C
diwstrt     EQU   $08E
diwstop     EQU   $090
ddfstrt     EQU   $092
ddfstop     EQU   $094
dmacon      EQU   $096
clxcon      EQU   $098
intena      EQU   $09A
intreq      EQU   $09C
adkcon      EQU   $09E

aud         EQU   $0A0
aud0        EQU   $0A0
aud1        EQU   $0B0
aud2        EQU   $0C0
aud3        EQU   $0D0

* AudChannel
ac_ptr      EQU   $00   ; ptr to start of waveform data
ac_len      EQU   $04   ; length of waveform in words
ac_per      EQU   $06   ; sample period
ac_vol      EQU   $08   ; volume
ac_dat      EQU   $0A   ; sample pair
ac_SIZEOF   EQU   $10

bplpt       EQU   $0E0

bplcon0     EQU   $100
bplcon1     EQU   $102
bplcon2     EQU   $104
bplcon3     EQU   $106
bpl1mod     EQU   $108
bpl2mod     EQU   $10A
bplhmod     EQU   $10C

bpldat      EQU   $110

sprpt       EQU   $120

spr         EQU   $140

* SpriteDef
sd_pos      EQU   $00
sd_ctl      EQU   $02
sd_dataa    EQU   $04
sd_dataB    EQU   $06
sd_SIZEOF   EQU   $08

color       EQU   $180

```

hardware/custom.i

```
htotal      EQU   $1c0
hsstop      EQU   $1c2
hbstrt      EQU   $1c4
hbstop      EQU   $1c6
vtotal      EQU   $1c8
vsstop      EQU   $1ca
vbstrt      EQU   $1cc
vbstop      EQU   $1ce
sprhstrt    EQU   $1d0
sprhstop    EQU   $1d2
bplhstrt    EQU   $1d4
bplhstop    EQU   $1d6
hhposw      EQU   $1d8
hhposr      EQU   $1da
beamcom0    EQU   $1dc
hsstrt      EQU   $1de
vsstrt      EQU   $1e0
hcenter     EQU   $1e2
diwhigh     EQU   $1e4

   ENDC  !HARDWARE_CUSTOM_I
```

hardware/dmabits.i

```
        IFND    HARDWARE_DMABITS_I
HARDWARE_DMABITS_I      SET     1
**
**      $Filename: hardware/dmabits.i $
**      $Release: 2.04 $
**      $Revision: 36.2 $
**      $Date: 90/07/10 $
**
**      include file for defining dma control stuff
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

* write definitions for dmaconw
DMAF_SETCLR     EQU   $8000
DMAF_AUDIO      EQU   $000F  * 4 bit mask
DMAF_AUD0       EQU   $0001
DMAF_AUD1       EQU   $0002
DMAF_AUD2       EQU   $0004
DMAF_AUD3       EQU   $0008
DMAF_DISK       EQU   $0010
DMAF_SPRITE     EQU   $0020
DMAF_BLITTER    EQU   $0040
DMAF_COPPER     EQU   $0080
DMAF_RASTER     EQU   $0100
DMAF_MASTER     EQU   $0200
DMAF_BLITHOG    EQU   $0400
DMAF_ALL        EQU   $01FF  * all dma channels

* read definitions for dmaconr
* bits 0-8 correspnd to dmaconw definitions
DMAF_BLTDONE    EQU   $4000
DMAF_BLTNZERO   EQU   $2000

DMAB_SETCLR     EQU   15
DMAB_AUD0       EQU   0
DMAB_AUD1       EQU   1
DMAB_AUD2       EQU   2
DMAB_AUD3       EQU   3
DMAB_DISK       EQU   4
DMAB_SPRITE     EQU   5
DMAB_BLITTER    EQU   6
DMAB_COPPER     EQU   7
DMAB_RASTER     EQU   8
DMAB_MASTER     EQU   9
DMAB_BLITHOG    EQU   10
DMAB_BLTDONE    EQU   14
DMAB_BLTNZERO   EQU   13

        ENDC    ; HARDWARE_DMABITS_I
```

hardware/intbits.i

```
        IFND    HARDWARE_INTBITS_I
HARDWARE_INTBITS_I      SET     1
**
**      $Filename: hardware/intbits.i $
**      $Release: 2.04 $
**      $Revision: 36.2 $
**      $Date: 90/07/10 $
**
**      bits in the interrupt enable (and interrupt request) register
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

INTB_SETCLR     EQU     (15)    ;Set/Clear control bit. Determines if bits
                                ;written with a 1 get set or cleared. Bits
                                ;written with a zero are allways unchanged.
INTB_INTEN      EQU     (14)    ;Master interrupt (enable only )
INTB_EXTER      EQU     (13)    ;External interrupt
INTB_DSKSYNC    EQU     (12)    ;Disk re-SYNChronized
INTB_RBF        EQU     (11)    ;serial port Receive Buffer Full
INTB_AUD3       EQU     (10)    ;Audio channel 3 block finished
INTB_AUD2       EQU     (9)     ;Audio channel 2 block finished
INTB_AUD1       EQU     (8)     ;Audio channel 1 block finished
INTB_AUD0       EQU     (7)     ;Audio channel 0 block finished
INTB_BLIT       EQU     (6)     ;Blitter finished
INTB_VERTB      EQU     (5)     ;start of Vertical Blank
INTB_COPER      EQU     (4)     ;Coprocessor
INTB_PORTS      EQU     (3)     ;I/O Ports and timers
INTB_SOFTINT    EQU     (2)     ;software interrupt request
INTB_DSKBLK     EQU     (1)     ;Disk Block done
INTB_TBE        EQU     (0)     ;serial port Transmit Buffer Empty



INTF_SETCLR     EQU     (1<<15)
INTF_INTEN      EQU     (1<<14)
INTF_EXTER      EQU     (1<<13)
INTF_DSKSYNC    EQU     (1<<12)
INTF_RBF        EQU     (1<<11)
INTF_AUD3       EQU     (1<<10)
INTF_AUD2       EQU     (1<<9)
INTF_AUD1       EQU     (1<<8)
INTF_AUD0       EQU     (1<<7)
INTF_BLIT       EQU     (1<<6)
INTF_VERTB      EQU     (1<<5)
INTF_COPER      EQU     (1<<4)
INTF_PORTS      EQU     (1<<3)
INTF_SOFTINT    EQU     (1<<2)
INTF_DSKBLK     EQU     (1<<1)
INTF_TBE        EQU     (1<<0)

        ENDC    ; HARDWARE_INTBITS_I
```

intuition/cghooks.i

```
    IFND INTUITION_CGHOOKS_I
INTUITION_CGHOOKS_I SET 1
**
** $Filename: intuition/cghooks.i $
** $Release: 2.04 $
** $Revision: 36.3 $
** $Date: 91/02/05 $
**
**  Custom gadget processing
**
**  (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**

    IFND EXEC_TYPES_I
    INCLUDE "exec/types.i"
    ENDC

    IFND INTUITION_INTUITION_I
    INCLUDE "intuition/intuition.i"
    ENDC

; =================================================================
; === Gadget Info =================================================
; =================================================================

; Package of information passed to custom and 'boopsi'
; gadget "hook" functions.  This structure is READ ONLY.

 STRUCTURE GadgetInfo,0
;    APTR ggi_Gadget

    APTR ggi_Screen
    APTR ggi_Window       ; null for screen gadgets
    APTR ggi_Requester    ; null if not GTYP_REQGADGET

    ; rendering information:
    ; don't use these without cloning/locking.
    ; Official way is to call ObtainRPort()
    APTR ggi_RastPort
    APTR ggi_Layer

    ; copy of dimensions of screen/window/g00/req/group
    ; that gadget resides in.  Left/Top of this box is
    ; offset from window mouse coordinates to gadget coordinates
    ;   screen gadgets:                 0,0 (from screen coords)
    ;   window gadgets (no g00):        0,0
    ;  GZZGADGETs (borderlayer):        0,0
    ;  GZZ innerlayer gadget:   borderleft, bordertop
    ;  Requester gadgets:               reqleft, reqtop
    STRUCT      ggi_Domain,ibox_SIZEOF

    ; these are the pens for the window or screen
    STRUCT      ggi_Pens,2      ; detail and block pen UBYTE's

    ; the Detail and Block pens in ggi_DrInfo->dri_Pens[] are
    ; for the screen. Use the above for window-sensitive
    ; colors.
    APTR ggi_DrInfo

    ; the size of this struct is not defined, since it is allocated
    ; ONLY by Intuition.
    ;    LABEL ggi_SIZEOF


    ENDC
```

intuition/classes.i

```
        IFND INTUITION_CLASSES_I
INTUITION_CLASSES_I SET 1
**
**  $Filename: intuition/classes.i $
**  $Release: 2.04 $
**  $Revision: 36.2 $
**  $Date: 90/11/02 $
**
**  Only used by class implementors
**
**  (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**

        IFND UTILITY_HOOKS_I
        INCLUDE "utility/hooks.i"
        ENDC

        IFND    INTUITION_CLASSUSR_I
        INCLUDE "intuition/classusr.i"
        ENDC

;*******************************************
;*** "White box" access to struct IClass ***
;*******************************************

 STRUCTURE ICLASS,0
    STRUCT cl_Dispatcher,h_SIZEOF
    ULONG  cl_Reserved           ; must be 0

    APTR   cl_Super
    APTR   cl_ID                 ; pointer to null-terminated string

    ; where within an object is the instance data for this class?
    UWORD  cl_InstOffset
    UWORD  cl_InstSize

    ULONG  cl_UserData           ; per-class data of your choice
    ULONG  cl_SubclassCount      ; how many direct subclasses?
    ULONG  cl_ObjectCount        ; how many objects created of this class?
    ULONG  cl_Flags
    ; no iclass_SIZEOF because only Intuition allocates these

; defined values of cl_Flags
CLB_INLIST EQU 0
CLF_INLIST EQU $00000001         ; class in in public class list

; see classes.h for common calculations (sorry, no macros yet)

;**********************************************
;*** "White box" access to struct _Object   ***
;**********************************************

* We have this, the instance data of the root class, PRECEDING
* the "object".  This is so that Gadget objects are Gadget pointers,
* and so on.  If this structure grows, it will always have o_Class
* at the end, so the you can always get it by subtracting #4 from
* the pointer returned from NewObject().
*
* This data structure is subject to change.  Do not use the o_Node
* embedded structure.


 STRUCTURE _Object,0
    STRUCT o_Node,MLN_SIZE
    APTR   o_Class
```

intuition/classes.i Page 2

```

    ; this value may change but difference between it and offset of o_Class
    ; will remain constant
    LABEL  _object_SIZEOF

    ENDC
```

intuition/classusr.i Page 1

```
    IFND INTUITION_CLASSUSR_I
INTUITION_CLASSUSR_I SET 1
**
** $Filename: intuition/classusr.i $
** $Release: 2.04 $
** $Revision: 36.3 $
** $Date: 91/02/05 $
**
** For application users of Intuition object classes
**
**  (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**



    IFND UTILITY_HOOKS_I
    INCLUDE "utility/hooks.i"
    ENDC

* beginning of "method message" passed to class dispatchers
 STRUCTURE Msg,0
    ULONG msg_MethodID
    ; method-specific data follows, some examples below

* For now, see the class id's for Intuition basic classes
* defined in classusr.h.  Sorry there aren't macros for the strings yet.

* dispatched method ID's
* NOTE: Applications should use Intuition entry points, not these,
* for NewObject, DisposeObject, SetAttrs, SetGadgetAttrs, and GetAttr.

    ENUM $101
    EITEM OM_NEW                ; 'object' parameter is "true class"
    EITEM OM_DISPOSE            ; delete self (no parameters)
    EITEM OM_SET                ; set attribute (list)
    EITEM OM_GET                ; return single attribute value
    EITEM OM_ADDTAIL            ; add self to a List
    EITEM OM_REMOVE             ; remove self from list (no parameters)
    EITEM OM_NOTIFY             ; send to self: notify dependents
    EITEM OM_UPDATE             ; notification message from someone
    EITEM OM_ADDMEMBER          ; used by various classes with lists
    EITEM OM_REMMEMBER          ; used by various classes with lists

* Parameter "Messages" passed to methods.
* NOTE: All of these parameter packets
* start off by the longword MethodID, but
* we don't redefine it for each structure.

* OM_NEW and OM_SET
 STRUCTURE opSet,4
    ; ULONG             MethodID
    APTR                ops_AttrList    ; new attributes
    APTR                ops_GInfo       ; always there for gadgets,
                                        ; but will be NULL for OM_NEW

* OM_NOTIFY, and OM_UPDATE
 STRUCTURE opUpdate,4
    ; ULONG             MethodID
    APTR                opu_AttrList    ; new attributes
    APTR                opu_GInfo       ; always there for gadgets,
                                        ; but will be NULL for OM_NEW
    ULONG               opu_Flags       ; defined below

* this flag means that the update message is being issued from
* something like an active gadget, ala GACT_FOLLOWMOUSE.  When
* the gadget goes inactive, it will issue a final update
* message with this bit cleared.  Examples of use are for
* GACT_FOLLOWMOUSE equivalents for propgadclass, and repeat strobes
* for buttons.

OPUB_INTERIM    EQU     0
OPUF_INTERIM    EQU     1

* OM_GET
 STRUCTURE opGet,4
    ; ULONG                 MethodID
    ULONG                   opg_AttrID
    APTR                    opg_Storage     ; may be other types, but "int"
                                            ; types are all ULONG

* OM_ADDTAIL
 STRUCTURE opAddTail,4
    ; ULONG                 MethodID
    APTR                    opat_List

* OM_ADDMEMBER, OM_REMMEMBER

 STRUCTURE opMember,4
    ; ULONG                 MethodID
    APTR                    opam_Object

 ENDC   ; IFND INTUITION_CLASSUSR_I
```

```
    IFND INTUITION_GADGETCLASS_I
INTUITION_GADGETCLASS_I SET 1
**
** $Filename: intuition/gadgetclass.i $
** $Release: 2.04 $
** $Revision: 36.8 $
** $Date: 91/02/22 $
**
** Custom and 'boopsi' gadget class interface
**
**  (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**      All Rights Reserved
**

        IFND EXEC_TYPES_I
        INCLUDE "exec/types.i"
        ENDC

        IFND INTUITION_INTUITION_I
        INCLUDE "intuition/intuition.i"
        ENDC

        IFND UTILITY_TAGITEM_I
        INCLUDE "utility/tagitem.i"
        ENDC


*
* NOTE:  intuition/iobsolete.i is included at the END of this file!
*

; Gadget Class attributes

GA_Dummy                EQU     (TAG_USER+$30000)
GA_Left                 EQU     (GA_Dummy+$0001)
GA_RelRight             EQU     (GA_Dummy+$0002)
GA_Top                  EQU     (GA_Dummy+$0003)
GA_RelBottom            EQU     (GA_Dummy+$0004)
GA_Width                EQU     (GA_Dummy+$0005)
GA_RelWidth             EQU     (GA_Dummy+$0006)
GA_Height               EQU     (GA_Dummy+$0007)
GA_RelHeight            EQU     (GA_Dummy+$0008)
GA_Text                 EQU     (GA_Dummy+$0009)   ; ti_Data is (UBYTE *)
GA_Image                EQU     (GA_Dummy+$000A)
GA_Border               EQU     (GA_Dummy+$000B)
GA_SelectRender         EQU     (GA_Dummy+$000C)
GA_Highlight            EQU     (GA_Dummy+$000D)
GA_Disabled             EQU     (GA_Dummy+$000E)
GA_GZZGadget            EQU     (GA_Dummy+$000F)
GA_ID                   EQU     (GA_Dummy+$0010)
GA_UserData             EQU     (GA_Dummy+$0011)
GA_SpecialInfo          EQU     (GA_Dummy+$0012)
GA_Selected             EQU     (GA_Dummy+$0013)
GA_EndGadget            EQU     (GA_Dummy+$0014)
GA_Immediate            EQU     (GA_Dummy+$0015)
GA_RelVerify            EQU     (GA_Dummy+$0016)
GA_FollowMouse          EQU     (GA_Dummy+$0017)
GA_RightBorder          EQU     (GA_Dummy+$0018)
GA_LeftBorder           EQU     (GA_Dummy+$0019)
GA_TopBorder            EQU     (GA_Dummy+$001A)
GA_BottomBorder         EQU     (GA_Dummy+$001B)
GA_ToggleSelect         EQU     (GA_Dummy+$001C)

* internal use only, until further notice, please
GA_SysGadget            EQU     (GA_Dummy+$001D)
* bool, sets GTYP_SYSGADGET field in type
```

```
GA_SysGType             EQU     (GA_Dummy+$001E)
* e.g., GTYP_WUPFRONT, ...

GA_Previous             EQU     (GA_Dummy+$001F)
* previous gadget (or (struct Gadget **)) in linked list
* NOTE: This attribute CANNOT be used to link new gadgets
* into the gadget list of an open window or requester.
* You must use AddGList().

GA_Next                 EQU     (GA_Dummy+$0020)
* not implemented

GA_DrawInfo             EQU     (GA_Dummy+$0021)
* some fancy gadgets need to see a DrawInfo
* when created or for layout

* You should use at most ONE of GA_Text, GA_IntuiText, and GA_LabelImage
GA_IntuiText            EQU     (GA_Dummy+$0022)
* ti_Data is (struct IntuiText  *)

GA_LabelImage           EQU     (GA_Dummy+$0023)
* ti_Data is an image (object), used in place of
* GadgetText

GA_TabCycle             EQU     (GA_Dummy+$0024)
* New for V37:
* Boolean indicates that this gadget is to participate in
* cycling activation with Tab or Shift-Tab.

* PROPGCLASS attributes

PGA_Dummy               EQU     (TAG_USER+$31000)
PGA_Freedom             EQU     (PGA_Dummy+$0001)
* either or both of FREEVERT and FREEHORIZ
PGA_Borderless          EQU     (PGA_Dummy+$0002)
PGA_HorizPot            EQU     (PGA_Dummy+$0003)
PGA_HorizBody           EQU     (PGA_Dummy+$0004)
PGA_VertPot             EQU     (PGA_Dummy+$0005)
PGA_VertBody            EQU     (PGA_Dummy+$0006)
PGA_Total               EQU     (PGA_Dummy+$0007)
PGA_Visible             EQU     (PGA_Dummy+$0008)
PGA_Top                 EQU     (PGA_Dummy+$0009)
; New for V37:
PGA_NewLook             EQU     (PGA_Dummy+$000A)

* STRGCLASS attributes

STRINGA_Dummy                   EQU     (TAG_USER+$32000)
STRINGA_MaxChars        EQU     (STRINGA_Dummy+$0001)
STRINGA_Buffer          EQU     (STRINGA_Dummy+$0002)
STRINGA_UndoBuffer      EQU     (STRINGA_Dummy+$0003)
STRINGA_WorkBuffer      EQU     (STRINGA_Dummy+$0004)
STRINGA_BufferPos       EQU     (STRINGA_Dummy+$0005)
STRINGA_DispPos         EQU     (STRINGA_Dummy+$0006)
STRINGA_AltKeyMap       EQU     (STRINGA_Dummy+$0007)
STRINGA_Font            EQU     (STRINGA_Dummy+$0008)
STRINGA_Pens            EQU     (STRINGA_Dummy+$0009)
STRINGA_ActivePens      EQU     (STRINGA_Dummy+$000A)
STRINGA_EditHook        EQU     (STRINGA_Dummy+$000B)
STRINGA_EditModes       EQU     (STRINGA_Dummy+$000C)

* booleans
STRINGA_ReplaceMode     EQU     (STRINGA_Dummy+$000D)
STRINGA_FixedFieldMode  EQU     (STRINGA_Dummy+$000E)
STRINGA_NoFilterMode    EQU     (STRINGA_Dummy+$000F)

```

```
STRINGA_Justification   EQU     (STRINGA_Dummy+$0010)
* GACT_STRINGCENTER, GACT_STRINGLEFT, GACT_STRINGRIGHT
STRINGA_LongVal         EQU     (STRINGA_Dummy+$0011)
STRINGA_TextVal         EQU     (STRINGA_Dummy+$0012)

STRINGA_ExitHelp        EQU     (STRINGA_Dummy+$0013)
* STRINGA_ExitHelp is new for V37, and ignored by V36.
* Set this if you want the gadget to exit when Help is
* pressed.  Look for a code of 0x5F, the rawkey code for Help

SG_DEFAULTMAXCHARS      EQU     (128)

* Gadget Layout related attributes

LAYOUTA_Dummy           EQU     (TAG_USER+$38000)
LAYOUTA_LayoutObj       EQU     (LAYOUTA_Dummy+$0001)
LAYOUTA_Spacing         EQU     (LAYOUTA_Dummy+$0002)
LAYOUTA_Orientation     EQU     (LAYOUTA_Dummy+$0003)

* orientation values
LORIENT_NONE            EQU     0
LORIENT_HORIZ           EQU     1
LORIENT_VERT            EQU     2

; Custom gadget hook command ID's
; (gadget class method/message ID's)

GM_HITTEST EQU          0       ; return GMR_GADGETHIT if you are clicked
                                ; (whether or not you are disabled)
GM_RENDER EQU           1       ; draw yourself, in the appropriate state
GM_GOACTIVE EQU         2       ; you are now going to be fed input
GM_HANDLEINPUT EQU      3       ; handle that input
GM_GOINACTIVE EQU       4       ; whether or not by choice, you are done

; Parameter "Messages" passed to gadget class methods

; All parameter structure begin with a MethodID field
; This definition of an abstract generic "message" is
; equivalent to a better one in intuition/classusr.i, but
; it's left here for historic reasons
 STRUCTURE MsgHeader,0
        ULONG   MethodID
        LABEL   methodid_SIZEOF

; GM_HITTEST
 STRUCTURE      gpHitTest,methodid_SIZEOF
    APTR        gpht_GInfo
    WORD        gpht_MouseX
    WORD        gpht_MouseY

; GM_HITTEST return value
GMR_GADGETHIT   EQU $00000004   ; if no hit, return 0

; GM_RENDER
 STRUCTURE      gpRender,methodid_SIZEOF
    APTR        gpr_GInfo       ; gadget context
    APTR        gpr_RPort       ; all ready for use
    LONG        gpr_Redraw      ; might be a "highlight pass"

; values of gpr_Redraw
GREDRAW_UPDATE  EQU 2   ; update for change in attributesvalues
GREDRAW_REDRAW  EQU 1   ; redraw gadget
GREDRAW_TOGGLE  EQU 0   ; toggle highlight, if applicable

; GM_GOACTIVE, GM_HANDLEINPUT
 STRUCTURE      gpInput,methodid_SIZEOF
```

intuition/gadgetclass.i

```
    APTR        gpi_GInfo;
    APTR        gpi_IEvent;
    APTR        gpi_Termination;
    WORD        gpi_MouseX
    WORD        gpi_MouseY

; GM_HANDLEINPUT and GM_GOACTIVE  return code flags
; return GMR_MEACTIVE (0) alone if you want more input.
; Otherwise, return ONE of GMR_NOREUSE and GMR_REUSE, and optionally
; GMR_VERIFY.

; here are the original constant "equates"
GMR_MEACTIVE     EQU $0000       ; (bugfix: was $0001 during beta)
GMR_NOREUSE      EQU $0002
GMR_REUSE        EQU $0004
GMR_VERIFY       EQU $0008       ; you MUST set cgp_Termination

* New for V37:
* You can end activation with one of GMR_NEXTACTIVE and GMR_PREVACTIVE,
* which instructs Intuition to activate the next or previous gadget
* that has GFLG_TABCYCLE set.
*
GMR_NEXTACTIVE   EQU $0010
GMR_PREVACTIVE   EQU $0020

; here are standard bit/flag pairs
GMRB_NOREUSE     EQU 1
GMRB_REUSE       EQU 2
GMRB_VERIFY      EQU 3
GMRB_NEXTACTIVE  EQU 4
GMRB_PREVACTIVE  EQU 5

GMRF_NOREUSE     EQU $0002
GMRF_REUSE       EQU $0004
GMRF_VERIFY      EQU $0008
GMRF_NEXTACTIVE  EQU $0010
GMRF_PREVACTIVE  EQU $0020

; GM_GOINACTIVE
 STRUCTURE      gpGoInactive,methodid_SIZEOF
    APTR        gpgi_GInfo;

* V37 field only!  DO NOT attempt to read under V36!
    ULONG       gpgi_Abort;      ; gpgi_Abort=1 if gadget was aborted
                                 ; by Intuition and 0 if gadget went
                                 ; inactive at its own request

* Include obsolete identifiers:
        IFND    INTUITION_IOBSOLETE_I
        INCLUDE "intuition/iobsolete.i"
        ENDC

        ENDC
```

intuition/icclass.i

```
    IFND INTUITION_ICCLASS_I
INTUITION_ICCLASS_I SET 1
**
**  $Filename: intuition/icclass.i $
**  $Release: 2.04 $
**  $Revision: 36.3 $
**  $Date: 91/02/05 $
**
**  Gadget/object interconnection classes
**
**  (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**


    IFND UTILITY_TAGITEM_I
    INCLUDE "utility/tagitem.i"
    ENDC

ICM_SETLOOP   EQU $402  ; set/increment loop counter
ICM_CLEARLOOP EQU $403  ; clear/decrement loop counter
ICM_CHECKLOOP EQU $404  ; set/increment loop

* no arguments for ICM_SETLOOP, ICM_CLEARLOOP, ICM_CHECKLOOP

* interconnection attributes used by icclass, modelclass, and gadgetclass

ICA_Dummy EQU           $40000
ICA_TARGET EQU          (ICA_Dummy+1)   ; interconnection target
ICA_MAP EQU             (ICA_Dummy+2)   ; interconnection map tagitem list
ICSPECIAL_CODE EQU      (ICA_Dummy+3)   ; a "pseudo-attribute",  see below.

* Normally, the value for ICA_TARGET is some object pointer,
* but if you specify the special value ICTARGET_IDCMP, notification
* will be send as an IDCMP_IDCMPUPDATE message to the appropriate window's
* IDCMP port.  See the definition of IDCMP_IDCMPUPDATE.
*
* When you specify ICTARGET_IDCMP for ICA_TARGET, the map you
* specify will be applied to derive the attribute list that is
* sent with the IDCMP_IDCMPUPDATE message.  If you specify a map list
* which results in the attribute tag id ICSPECIAL_CODE, the
* lower sixteen bits of the corresponding ti_Data value will
* be copied into the Code field of the IDCMP_IDCMPUPDATE IntuiMessage.

ICTARGET_IDCMP  EQU $ffffffff

    ENDC
    IFND INTUITION_IMAGECLASS_I
INTUITION_IMAGECLASS_I SET 1
**
**  $Filename: intuition/imageclass.i $
**  $Release: 2.04 $
**  $Revision: 36.3 $
**  $Date: 91/02/05 $
**
**  Definitions for the image classes
**
**  (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**

    IFND UTILITY_TAGITEM_I
    INCLUDE "utility/tagitem.i"
    ENDC

*
* NOTE:  intuition/iobsolete.i is included at the END of this file!
*

CUSTOMIMAGEDEPTH        EQU (-1)
* if image.Depth is this, it's a new Image class object

*****************************************************
IMAGE_ATTRIBUTES        EQU (TAG_USER+$20000)

IA_Left         EQU (IMAGE_ATTRIBUTES+$0001)
IA_Top          EQU (IMAGE_ATTRIBUTES+$0002)
IA_Width        EQU (IMAGE_ATTRIBUTES+$0003)
IA_Height       EQU (IMAGE_ATTRIBUTES+$0004)
IA_FGPen        EQU (IMAGE_ATTRIBUTES+$0005)
                    ; IA_FGPen also means "PlanePick"
IA_BGPen        EQU (IMAGE_ATTRIBUTES+$0006)
                    ; IA_BGPen also means "PlaneOnOff"
IA_Data         EQU (IMAGE_ATTRIBUTES+$0007)
                    ; bitplanes, for classic image,
                    ; other image classes may use it for other things
IA_LineWidth    EQU (IMAGE_ATTRIBUTES+$0008)
IA_Pens         EQU (IMAGE_ATTRIBUTES+$000E)
                    ; pointer to UWORD pens[],
                    ; ala DrawInfo.Pens, MUST be
                    ; terminated by ~0.  Some classes can
                    ; choose to have this, or SYSIA_DrawInfo,
                    ; or both.
IA_Resolution   EQU (IMAGE_ATTRIBUTES+$000F)
                    ; packed uwords for x/y resolution into a longword
                    ; ala DrawInfo.Resolution

* see class documentation to learn which
* classes recognize these
IA_APattern     EQU (IMAGE_ATTRIBUTES+$0010)
IA_APatSize     EQU (IMAGE_ATTRIBUTES+$0011)
IA_Mode         EQU (IMAGE_ATTRIBUTES+$0012)
IA_Font         EQU (IMAGE_ATTRIBUTES+$0013)
IA_Outline      EQU (IMAGE_ATTRIBUTES+$0014)
IA_Recessed     EQU (IMAGE_ATTRIBUTES+$0015)
IA_DoubleEmboss EQU (IMAGE_ATTRIBUTES+$0016)
IA_EdgesOnly    EQU (IMAGE_ATTRIBUTES+$0017)

* "sysiclass" attributes
SYSIA_Size      EQU (IMAGE_ATTRIBUTES+$000B)
                    ; EQU's below
SYSIA_Depth     EQU (IMAGE_ATTRIBUTES+$000C)
                    ; this is unused by Intuition.  SYSIA_DrawInfo
```

```
                    ; is used instead for V36
SYSIA_Which     EQU (IMAGE_ATTRIBUTES+$000D)
                    ; see EQU's below
SYSIA_DrawInfo  EQU (IMAGE_ATTRIBUTES+$0018)
                    ; pass to sysiclass, please

*  obsolete: don't use these, use IA_Pens
SYSIA_Pens      EQU IA_Pens
IA_ShadowPen    EQU (IMAGE_ATTRIBUTES+$0009)
IA_HighlightPen EQU (IMAGE_ATTRIBUTES+$000A)

* next attribute: (IMAGE_ATTRIBUTES+$0019)
*****************************************************

* data values for SYSIA_Size
SYSISIZE_MEDRES EQU (0)
SYSISIZE_LOWRES EQU (1)
SYSISIZE_HIRES  EQU (2)

*
* SYSIA_Which tag data values:
* Specifies which system gadget you want an image for.
* Some numbers correspond to internal Intuition #defines
DEPTHIMAGE      EQU ($00)
ZOOMIMAGE       EQU ($01)
SIZEIMAGE       EQU ($02)
CLOSEIMAGE      EQU ($03)
SDEPTHIMAGE     EQU ($05)              ; screen depth gadget
LEFTIMAGE       EQU ($0A)
UPIMAGE         EQU ($0B)
RIGHTIMAGE      EQU ($0C)
DOWNIMAGE       EQU ($0D)
CHECKIMAGE      EQU ($0E)
MXIMAGE         EQU ($0F)       ; mutual exclude "button"

* image message id's
IM_DRAW         EQU ($0202)     ; draw yourself, with "state"
IM_HITTEST      EQU ($0203)     ; return TRUE if click hits image
IM_ERASE        EQU ($0204)     ; erase yourself
IM_MOVE         EQU ($0205)     ; draw new and erase old, smoothly

IM_DRAWFRAME    EQU ($0206)     ; draw with specified dimensions
IM_FRAMEBOX     EQU ($0207)     ; get recommended frame around some box
IM_HITFRAME     EQU ($0208)     ; hittest with dimensions
IM_ERASEFRAME   EQU ($0209)     ; hittest with dimensions

* image draw states or styles, for IM_DRAW
IDS_NORMAL              EQU (0)
IDS_SELECTED            EQU (1) ; for selected gadgets
IDS_DISABLED            EQU (2) ; for disabled gadgets
IDS_BUSY                EQU (3) ; for future functionality
IDS_INDETERMINATE       EQU (4) ; for future functionality
IDS_INACTIVENORMAL      EQU (5) ; normal, in inactive window border
IDS_INACTIVESELECTED    EQU (6) ; selected, in inactive border
IDS_INACTIVEDISABLED    EQU (7) ; disabled, in inactive border

* oops, please forgive spelling error by jimm
IDS_INDETERMINANT EQU IDS_INDETERMINATE

* IM_FRAMEBOX
 STRUCTURE impFrameBox,4                ; starts with ULONG MethodID
    APTR        impf_ContentsBox        ; input: relative box of contents
    APTR        impf_FrameBox   ; output: rel. box of encl frame
    APTR        impf_DrInfo
    LONG        impf_FrameFlags

```

intuition/imageclass.i Page 3

```
; Make do with the dimensions of FrameBox provided.
FRAMEB_SPECIFY  EQU (0)
FRAMEF_SPECIFY  EQU (1)


* IM_DRAW, IM_DRAWFRAME
 STRUCTURE impDraw,4            ; starts with ULONG MethodID
    APTR        impd_RPort
    WORD        impd_OffsetX
    WORD        impd_OffsetY
    ULONG       impd_State
    APTR        impd_DrInfo
    ; these parameters only valid for IM_DRAWFRAME
    WORD        impd_DimensionsWidth
    WORD        impd_DimensionsHeight

* IM_ERASE, IM_ERASEFRAME
* NOTE: This is a subset of impDraw
 STRUCTURE impErase,4           ; starts with ULONG MethodID
    APTR        impe_RPort
    WORD        impe_OffsetX
    WORD        impe_OffsetY
    ; these parameters only valid for IM_ERASEFRAME
    WORD        impe_DimensionsWidth
    WORD        impe_DimensionsHeight

* IM_HITTEST, IM_HITFRAME
 STRUCTURE impHitTest,4         ; starts with ULONG MethodID
    ; these parameters only valid for IM_ERASEFRAME
    WORD        imph_PointX
    WORD        imph_PointY
    ; these parameters only valid for IM_HITFRAME
    WORD        imph_DimensionsWidth
    WORD        imph_DimensionsHeight


* Include obsolete identifiers:
        IFND    INTUITION_IOBSOLETE_I
        INCLUDE "intuition/iobsolete.i"
        ENDC

    ENDC
```

intuition/intuition.i Page 1

```
        IFND    INTUITION_INTUITION_I
INTUITION_INTUITION_I   SET     1
**
**      $Filename: intuition/intuition.i $
**      $Release: 2.04 $
**      $Revision: 36.28 $
**      $Date: 91/02/22 $
**
**      Interface definitions for Intuition applications
**
**      (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**

        IFND EXEC_TYPES_I
        INCLUDE "exec/types.i"
        ENDC

        IFND    GRAPHICS_GFX_I
        INCLUDE "graphics/gfx.i"
        ENDC

        IFND    GRAPHICS_CLIP_I
        INCLUDE "graphics/clip.i"
        ENDC

        IFND    GRAPHICS_VIEW_I
        INCLUDE "graphics/view.i"
        ENDC

        IFND    GRAPHICS_RASTPORT_I
        INCLUDE "graphics/rastport.i"
        ENDC

        IFND    GRAPHICS_LAYERS_I
        INCLUDE "graphics/layers.i"
        ENDC

        IFND    GRAPHICS_TEXT_I
        INCLUDE "graphics/text.i"
        ENDC

        IFND EXEC_PORTS_I
        INCLUDE "exec/ports.i"
        ENDC

        IFND    DEVICES_TIMER_I
        INCLUDE "devices/timer.i"
        ENDC

        IFND    DEVICES_INPUTEVENT_I
        INCLUDE "devices/inputevent.i"
        ENDC

        IFND UTILITY_TAGITEM_I
        INCLUDE "utility/tagitem.i"
        ENDC

*
* NOTE:  intuition/iobsolete.i is included at the END of this file!
*


; =========================================================================;
; === Menu ================================================================;
; =========================================================================;
```

intuition/intuition.i Page 2

```
 STRUCTURE Menu,0

    APTR  mu_NextMenu    ; menu pointer, same level
    WORD mu_LeftEdge     ; position of the select box
    WORD mu_TopEdge      ; position of the select box
    WORD mu_Width        ; dimensions of the select box
    WORD mu_Height       ; dimensions of the select box
    WORD mu_Flags        ; see flag definitions below
    APTR mu_MenuName     ; text for this Menu Header
    APTR  mu_FirstItem   ; pointer to first in chain

    ; these mysteriously-named variables are for internal use only
    WORD mu_JazzX
    WORD mu_JazzY
    WORD mu_BeatX
    WORD mu_BeatY

    LABEL mu_SIZEOF

;*** FLAGS SET BY BOTH THE APPLIPROG AND INTUITION ***
MENUENABLED EQU $0001    ; whether or not this menu is enabled

;*** FLAGS SET BY INTUITION ***
MIDRAWN EQU $0100        ; this menu's items are currently drawn

; ====================================================================;
; === MenuItem =======================================================;
; ====================================================================;
 STRUCTURE MenuItem,0

    APTR mi_NextItem     ; pointer to next in chained list
    WORD mi_LeftEdge     ; position of the select box
    WORD mi_TopEdge      ; position of the select box
    WORD mi_Width        ; dimensions of the select box
    WORD mi_Height       ; dimensions of the select box
    WORD mi_Flags        ; see the defines below

    LONG mi_MutualExclude ; set bits mean this item excludes that item

    APTR mi_ItemFill     ; points to Image, IntuiText, or NULL

    ; when this item is pointed to by the cursor and the items highlight
    ; mode HIGHIMAGE is selected, this alternate image will be displayed
    APTR mi_SelectFill   ; points to Image, IntuiText, or NULL

    BYTE mi_Command      ; only if appliprog sets the COMMSEQ flag

    BYTE mi_KludgeFill00 ; This is strictly for word-alignment

    APTR mi_SubItem      ; if non-zero, points to MenuItem for submenu

   ; The NextSelect field represents the menu number of next selected
   ; item (when user has drag-selected several items)
    WORD mi_NextSelect

    LABEL  mi_SIZEOF

; --- FLAGS SET BY THE APPLIPROG ----------------------------------------------
CHECKIT         EQU $0001       ; set to indicate checkmarkable item
ITEMTEXT        EQU $0002       ; set if textual, clear if graphical item
COMMSEQ         EQU $0004       ; set if there's an command sequence
MENUTOGGLE      EQU $0008       ; set for toggling checks (else mut. exclude)
ITEMENABLED     EQU $0010       ; set if this item is enabled

; these are the SPECIAL HIGHLIGHT FLAG state meanings
HIGHFLAGS       EQU $00C0       ; see definitions below for these bits
```

intuition/intuition.i Page 3

```
HIGHIMAGE       EQU $0000       ; use the user's "select image"
HIGHCOMP        EQU $0040       ; highlight by complementing the select box
HIGHBOX         EQU $0080       ; highlight by drawing a box around the image
HIGHNONE        EQU $00C0       ; don't highlight

; --- FLAGS SET BY BOTH APPLIPROG AND INTUITION ----------------------------
CHECKED EQU $0100       ; state of the checkmark


; --- FLAGS SET BY INTUITION -----------------------------------------------
ISDRAWN         EQU $1000       ; this item's subs are currently drawn
HIGHITEM        EQU $2000       ; this item is currently highlighted
MENUTOGGLED     EQU $4000       ; this item was already toggled






; ===================================================================
; === Requester =====================================================
; ===================================================================
 STRUCTURE Requester,0

    APTR  rq_OlderRequest
    WORD rq_LeftEdge            ; dimensions of the entire box
    WORD rq_TopEdge             ; dimensions of the entire box
    WORD rq_Width               ; dimensions of the entire box
    WORD rq_Height              ; dimensions of the entire box

    WORD rq_RelLeft             ; get POINTREL Pointer relativity offsets
    WORD rq_RelTop              ; get POINTREL Pointer relativity offsets

    APTR  rq_ReqGadget          ; pointer to the first of a list of gadgets
    APTR  rq_ReqBorder          ; the box's border
    APTR  rq_ReqText            ; the box's text

    WORD  rq_Flags              ; see definitions below

    UBYTE rq_BackFill           ; pen number for back-plane fill before draws

    BYTE rq_KludgeFill00        ; This is strictly for word-alignment

    APTR rq_ReqLayer            ; layer in which requester rendered
    STRUCT rq_ReqPad1,32        ; for backwards compatibility (reserved)

    ; If the BitMap plane pointers are non-zero, this tells the system
    ; that the image comes pre-drawn (if the appliprog wants to define
    ; its own box, in any shape or size it wants!); this is OK by
    ; Intuition as long as there's a good correspondence between the image
    ; and the specified Gadgets
    APTR  rq_ImageBMap          ; points to the BitMap of PREDRAWN imagery

    APTR  rq_RWindow            ; points back to requester's window
    APTR  rq_ReqImage           ; new for V36: drawn if USEREQIMAGE set
    STRUCT rq_ReqPad2,32        ; for backwards compatibility (reserved)

    LABEL rq_SIZEOF

; FLAGS SET BY THE APPLIPROG
POINTREL        EQU $0001  ; if POINTREL set, TopLeft is relative to pointer
                           ; for DMRequester, relative to window center
                           ; for Request().
PREDRAWN        EQU $0002  ; if ReqBMap points to predrawn Requester imagery
NOISYREQ        EQU $0004  ; if you don't want requester to filter input

```

```
; New for V36
SIMPLEREQ       EQU $0010  ; to use SIMPLEREFRESH layer (recommended)
USEREQIMAGE     EQU $0020  ; render linked list ReqImage after BackFill
                           ; but before gadgets and text
NOREQBACKFILL   EQU $0040  ; don't bother filling with Requester.BackFill


; FLAGS SET BY INTUITION;
REQOFFWINDOW    EQU $1000       ; part of one of the Gadgets was offwindow
REQACTIVE       EQU $2000       ; this requester is active
SYSREQUEST      EQU $4000       ; this requester caused by system
DEFERREFRESH    EQU $8000       ; this Requester stops a Refresh broadcast





; ======================================================================
; === Gadget ===========================================================
; ======================================================================
 STRUCTURE Gadget,0

    APTR gg_NextGadget          ; next gadget in the list

    WORD gg_LeftEdge            ; "hit box" of gadget
    WORD gg_TopEdge             ; "hit box" of gadget
    WORD gg_Width               ; "hit box" of gadget
    WORD gg_Height              ; "hit box" of gadget

    WORD gg_Flags               ; see below for list of defines

    WORD gg_Activation          ; see below for list of defines

    WORD gg_GadgetType          ; see below for defines

    ; appliprog can specify that the Gadget be rendered as either as Border
    ; or an Image.  This variable points to which (or equals NULL if there's
    ; nothing to be rendered about this Gadget)
    APTR gg_GadgetRender

    ; appliprog can specify "highlighted" imagery rather than algorithmic
    ; this can point to either Border or Image data
    APTR gg_SelectRender

    APTR gg_GadgetText          ; text for this gadget;

    ; MutualExclude, never implemented, is now declared obsolete.
    ; There are published examples of implementing a more general
    ; and practical exclusion in your applications.
    ;
    ; Starting V36, this field is used to point to a hook
    ; for a custom gadget.
    ;
    ; Programs using this field for their own processing will
    ; continue to work, as long as they don't try the
    ; trick with custom gadgets
    LONG gg_MutualExclude       ; obsolete

    ; pointer to a structure of special data required by Proportional, String
    ; and Integer Gadgets
    APTR gg_SpecialInfo

    WORD gg_GadgetID    ; user-definable ID field
    APTR  gg_UserData   ; ptr to general purpose User data (ignored by Intuit)

    LABEL gg_SIZEOF
```

```

; --- Gadget.Flags values ---
; combinations in these bits describe the highlight technique to be used
GFLG_GADGHIGHBITS           EQU $0003
GFLG_GADGHCOMP              EQU $0000       ; Complement the select box
GFLG_GADGHBOX               EQU $0001       ; Draw a box around the image
GFLG_GADGHIMAGE             EQU $0002       ; Blast in this alternate image
GFLG_GADGHNONE              EQU $0003       ; don't highlight

; set this flag if the GadgetRender and SelectRender point to Image imagery,
; clear if it's a Border
GFLG_GADGIMAGE              EQU $0004

; combinations in these next two bits specify to which corner the gadget's
; Left & Top coordinates are relative.  If relative to Top/Left,
; these are "normal" coordinates (everything is relative to something in
; this universe)
GFLG_RELBOTTOM              EQU $0008       ; set if rel to bottom, clear if to top
GFLG_RELRIGHT               EQU $0010       ; set if rel to right, clear if to left
; set the GFLG_RELWIDTH bit to spec that Width is relative to width of screen
GFLG_RELWIDTH               EQU $0020
; set the GFLG_RELHEIGHT bit to spec that Height is rel to height of screen
GFLG_RELHEIGHT              EQU $0040

; the GFLG_SELECTED flag is initialized by you and set by Intuition.  It
; specifies whether or not this Gadget is currently selected/highlighted
GFLG_SELECTED               EQU $0080


; the GFLG_DISABLED flag is initialized by you and later set by Intuition
; according to your calls to On/OffGadget().  It specifies whether or not
; this Gadget is currently disabled from being selected
GFLG_DISABLED               EQU $0100

* These flags specify the type of text field that Gadget.GadgetText
* points to.  In all normal (pre-V36) gadgets which you initialize
* this field should always be zero.  Some types of gadget objects
* created from classes will use these fields to keep track of
* types of labels/contents that different from IntuiText, but are
* stashed in GadgetText.

GFLG_LABELMASK              EQU     $3000
GFLG_LABELITEXT             EQU     $0000   ; GadgetText points to IntuiText
GFLG_LABELSTRING            EQU     $1000   ; GadgetText points to (UBYTE *)
GFLG_LABELIMAGE             EQU     $2000   ; GadgetText points to Image (object)
; New for V37: GFLG_TABCYCLE
GFLG_TABCYCLE               EQU     $0200   ; (string or custom) gadget
                                            ; participates in cycling activation with
                                            ; Tab or Shift-Tab

; New for V37: GFLG_STRINGEXTEND.  We discovered that V34 doesn't properly
; ignore the value we had chosen for the Gadget->Activation flag
; GACT_STRINGEXTEND.  NEVER SET THAT FLAG WHEN RUNNING UNDER V34.
; The Gadget->Flags bit GFLG_STRINGEXTEND is provided as a synonym which is
; safe under V34, and equivalent to GACT_STRINGEXTEND under V37.
; (Note that the two flags are not numerically equal)
GFLG_STRINGEXTEND           EQU     $0400   ; this String Gadget has StringExtend



; --- These are the Activation flag bits ----------------------------------
; GACT_RELVERIFY is set if you want to verify that the pointer was still over
; the gadget when the select button was released.  Will cause
; an IDCMP_GADGETUP message to be sent if so.
GACT_RELVERIFY              EQU $0001

```

```
; the flag GACT_IMMEDIATE, when set, informs the caller that the gadget
; was activated when it was activated.  this flag works in conjunction with
; the GACT_RELVERIFY flag
GACT_IMMEDIATE          EQU $0002

; the flag GACT_ENDGADGET, when set, tells the system that this gadget, when
; selected, causes the Requester or AbsMessage to be ended.  Requesters or
; AbsMessages that are ended are erased and unlinked from the system
GACT_ENDGADGET          EQU $0004

; the GACT_FOLLOWMOUSE flag, when set, specifies that you want to receive
; reports on mouse movements while this gadget is active.
; You probably want to set the GACT_IMMEDIATE flag when using
; GACT_FOLLOWMOUSE, since that's the only reasonable way you have of learning
; why Intuition is suddenly sending you a stream of mouse movement events.
; If you don't set GACT_RELVERIFY, you'll get at least one Mouse Position
; event.
GACT_FOLLOWMOUSE        EQU $0008

; if any of the BORDER flags are set in a Gadget that's included in the
; Gadget list when a Window is opened, the corresponding Border will
; be adjusted to make room for the Gadget
GACT_RIGHTBORDER        EQU $0010
GACT_LEFTBORDER         EQU $0020
GACT_TOPBORDER          EQU $0040
GACT_BOTTOMBORDER       EQU $0080
GACT_BORDERSNIFF        EQU $8000       ; neither set nor rely on this bit

GACT_TOGGLESELECT       EQU $0100       ; this bit for toggle-select mode
GACT_BOOLEXTEND         EQU $2000       ; This Boolean Gadget has a BoolInfo

; should properly be in StringInfo, but aren't
GACT_STRINGLEFT         EQU $0000       ; NOTE WELL: that this has value zero
GACT_STRINGCENTER       EQU $0200       ; center the String
GACT_STRINGRIGHT        EQU $0400       ; right-justify the String
GACT_LONGINT            EQU $0800       ; This String Gadget is a Long Integer
GACT_ALTKEYMAP          EQU $1000       ; has an alternate keymapping
GACT_STRINGEXTEND       EQU $2000       ; this String Gadget has an extension
                                        ; NOTE: NEVER SET GACT_STRINGEXTEND
                                        ; IF YOU ARE RUNNING ON LESS THAN V36!
                                        ; SEE GFLG_STRINGEXTEND (ABOVE) INSTEAD

GACT_ACTIVEGADGET       EQU $4000       ; this gadget is "active".  This flag
                                ; is maintained by Intuition, and you
                                ; cannot count on its value persisting
                                ; while you do something on your program's
                                ; task.  It can only be trusted by
                                ; people implementing custom gadgets

* note $8000 is used above (GACT_BORDERSNIFF); all Activation flags defined



; --- GADGET TYPES -------------------------------------------------------
; These are the Gaget Type definitions for the variable GadgetType.
; Gadget number type MUST start from one.  NO TYPES OF ZERO ALLOWED.
; first comes the mask for Gadget flags reserved for Gadget typing
GTYP_GADGETTYPE         EQU $FC00       ; all Gadget Global Type flags
                                        ; (padded)
GTYP_SYSGADGET          EQU $8000       ; 1 = Allocated by the system, 0 = by ap
  p.
GTYP_SCRGADGET          EQU $4000       ; 1 = ScreenGadget, 0 = WindowGadget
GTYP_GZZGADGET          EQU $2000       ; 1 = Gadget for WFLG_GIMMEZEROZERO
                                        ;     borders
GTYP_REQGADGET          EQU $1000       ; 1 = this is a Requester Gadget
; system gadgets
```

```
GTYP_SIZING             EQU $0010
GTYP_WDRAGGING          EQU $0020
GTYP_SDRAGGING          EQU $0030
GTYP_WUPFRONT           EQU $0040
GTYP_SUPFRONT           EQU $0050
GTYP_WDOWNBACK          EQU $0060
GTYP_SDOWNBACK          EQU $0070
GTYP_CLOSE              EQU $0080
; application gadgets
GTYP_BOOLGADGET         EQU $0001
GTYP_GADGET0002         EQU $0002
GTYP_PROPGADGET         EQU $0003
GTYP_STRGADGET          EQU $0004
GTYP_CUSTOMGADGET       EQU $0005
GTYP_GTYPEMASK          EQU $0004       ; masks out to gadget class



; ======================================================================
; === BoolInfo==========================================================
; ======================================================================
; This is the special data needed by an Extended Boolean Gadget
; Typically this structure will be pointed to by the Gadget field SpecialInfo

 STRUCTURE BoolInfo,0

    WORD    bi_Flags    ; defined below
    APTR    bi_Mask     ; bit mask for highlighting and selecting
                        ; mask must follow the same rules as an Image
                        ; plane.  Its width and height are determined
                        ; by the width and height of the gadget's
                        ; select box. (i.e. Gadget.Width and .Height).
    LONG    bi_Reserved ; set to 0

    LABEL   bi_SIZEOF

; set BoolInfo.Flags to this flag bit.
; in the future, additional bits might mean more stuff hanging
; off of BoolInfo.Reserved.

BOOLMASK        EQU     $0001   ; extension is for masked gadget

; ======================================================================
; === PropInfo =========================================================
; ======================================================================
; this is the special data required by the proportional Gadget
; typically, this data will be pointed to by the Gadget variable SpecialInfo
 STRUCTURE PropInfo,0

    WORD pi_Flags       ; general purpose flag bits (see defines below)

    ; You initialize the Pot variables before the Gadget is added to
    ; the system.  Then you can look here for the current settings
    ; any time, even while User is playing with this Gadget.  To
    ; adjust these after the Gadget is added to the System, use
    ; ModifyProp(); The Pots are the actual proportional settings,
    ; where a value of zero means zero and a value of MAXPOT means
    ; that the Gadget is set to its maximum setting.
    WORD pi_HorizPot    ; 16-bit FixedPoint horizontal quantity percentage;
    WORD pi_VertPot     ; 16-bit FixedPoint vertical quantity percentage;

    ; the 16-bit FixedPoint Body variables describe what percentage
    ; of the entire body of stuff referred to by this Gadget is
    ; actually shown at one time.  This is used with the AUTOKNOB
    ; routines, to adjust the size of the AUTOKNOB according to how
    ; much of the data can be seen.  This is also used to decide how
    ; far to advance the Pots when User hits the Container of the Gadget.
    ; For instance, if you were controlling the display of a 5-line
    ; Window of text with this Gadget, and there was a total of 15
    ; lines that could be displayed, you would set the VertBody value to
    ;    (MAXBODY / (TotalLines / DisplayLines)) = MAXBODY / 3.
    ; Therefore, the AUTOKNOB would fill 1/3 of the container, and if
    ; User hits the Cotainer outside of the knob, the pot would advance
    ; 1/3 (plus or minus) If there's no body to show, or the total
    ; amount of displayable info is less than the display area, set the
    ; Body variables to the MAX.  To adjust these after the Gadget is
    ; added to the System, use ModifyProp().
    WORD pi_HorizBody   ; horizontal Body
    WORD pi_VertBody    ; vertical Body

    ; these are the variables that Intuition sets and maintains
    WORD pi_CWidth      ; Container width (with any relativity absoluted)
    WORD pi_CHeight     ; Container height (with any relativity absoluted)
    WORD pi_HPotRes     ; pot increments
    WORD pi_VPotRes     ; pot increments
    WORD pi_LeftBorder  ; Container borders
    WORD pi_TopBorder   ; Container borders
    LABEL  pi_SIZEOF

; --- FLAG BITS -------------------------------------------------------------
AUTOKNOB        EQU $0001       ; this flag sez:  gimme that old auto-knob
FREEHORIZ       EQU $0002       ; if set, the knob can move horizontally
FREEVERT        EQU $0004       ; if set, the knob can move vertically
PROPBORDERLESS  EQU $0008       ; if set, no border will be rendered
KNOBHIT         EQU $0100       ; set when this Knob is hit
PROPNEWLOOK     EQU $0010       ; set this if you want to get the new
                                ; V36 look

KNOBHMIN        EQU 6           ; minimum horizontal size of the knob
KNOBVMIN        EQU 4           ; minimum vertical size of the knob
MAXBODY         EQU $FFFF       ; maximum body value
MAXPOT          EQU $FFFF       ; maximum pot value


; ========================================================================
; === StringInfo =========================================================
; ========================================================================
; this is the special data required by the string Gadget
; typically, this data will be pointed to by the Gadget variable SpecialInfo
 STRUCTURE StringInfo,0

    ; you initialize these variables, and then Intuition maintains them
    APTR  si_Buffer     ; the buffer containing the start and final string
    APTR  si_UndoBuffer ; optional buffer for undoing current entry
    WORD si_BufferPos   ; character position in Buffer
    WORD si_MaxChars    ; max number of chars in Buffer (including NULL)
    WORD si_DispPos     ; Buffer position of first displayed character

    ; Intuition initializes and maintains these variables for you
    WORD si_UndoPos     ; character position in the undo buffer
    WORD si_NumChars    ; number of characters currently in Buffer
    WORD si_DispCount   ; number of whole characters visible in Container
    WORD si_CLeft       ; topleft offset of the container
    WORD si_CTop        ; topleft offset of the container

    ; unused field is changed to allow extended specification
    ; of string gadget parameters.  It is ignored unless the flag
    ; GACT_STRINGEXTEND is set in the Gadget's Activation field
    ; or the GFLG_STRINGEXTEND flag is set in the Gadget Flags field.
    ; (See GFLG_STRINGEXTEND for an important note)
    ;APTR  si_LayerPtr  ; --- obsolete ---
    APTR  si_Extension
```

```

    ; you can initialize this variable before the gadget is submitted to
    ; Intuition, and then examine it later to discover what integer
    ; the user has entered (if the user never plays with the gadget,
    ; the value will be unchanged from your initial setting)
    LONG  si_LongInt    ; the LONG return value of a GACT_LONGINT String Gad.

    ; If you want this Gadget to use your own Console keymapping, you
    ; set the GACT_ALTKEYMAP bit in the Activation flags of the Gadget, and
    ; then set this variable to point to your keymap.  If you don't set the
    ; GACT_ALTKEYMAP, you'll get the standard ASCII keymapping.
    APTR si_AltKeyMap

    LABEL si_SIZEOF




; ========================================================================
; === IntuiText ==========================================================
; ========================================================================
; IntuiText is a series of strings that start with a location
;  (always relative to the upper-left corner of something) and then the
; text of the string.  The text is null-terminated.
 STRUCTURE IntuiText,0

    BYTE it_FrontPen            ; the pens for rendering the text
    BYTE it_BackPen             ; the pens for rendering the text

    BYTE it_DrawMode            ; the mode for rendering the text

    BYTE it_KludgeFill00        ; This is strictly for word-alignment

    WORD it_LeftEdge            ; relative start location for the text
    WORD it_TopEdge             ; relative start location for the text

    APTR  it_ITextFont          ; if NULL, you accept the defaults

    APTR it_IText               ; pointer to null-terminated text

    APTR  it_NextText           ; pointer to another IntuiText to render

    LABEL it_SIZEOF





; ========================================================================
; === Border =============================================================
; ========================================================================
; Data type Border, used for drawing a series of lines which is intended for
; use as a border drawing, but which may, in fact, be used to render any
; arbitrary vector shape.
; The routine DrawBorder sets up the RastPort with the appropriate
; variables, then does a Move to the first coordinate, then does Draws
; to the subsequent coordinates.
; After all the Draws are done, if NextBorder is non-zero we call DrawBorder
; on NextBorder
 STRUCTURE Border,0

    WORD  bd_LeftEdge           ; initial offsets from the origin
    WORD  bd_TopEdge            ; initial offsets from the origin
    BYTE  bd_FrontPen           ; pen number for rendering
    BYTE  bd_BackPen            ; pen number for rendering
    BYTE  bd_DrawMode           ; mode for rendering
    BYTE  bd_Count                  ; number of XY pairs
    APTR  bd_XY                     ; vector coordinate pairs rel to LeftTop
    APTR  bd_NextBorder             ; pointer to any other Border too

    LABEL bd_SIZEOF


; ========================================================================
; === Image ==============================================================
; ========================================================================
; This is a brief image structure for very simple transfers of
; image data to a RastPort
 STRUCTURE Image,0

    WORD ig_LeftEdge                ; starting offset relative to something
    WORD ig_TopEdge                 ; starting offset relative to something
    WORD ig_Width                   ; pixel size (though data is word-aligned)
    WORD ig_Height                  ; pixel size
    WORD ig_Depth                   ; pixel size
    APTR ig_ImageData               ; pointer to the actual image bits

    ; the PlanePick and PlaneOnOff variables work much the same way as the
    ; equivalent GELS Bob variables.  It's a space-saving
    ; mechanism for image data.  Rather than defining the image data
    ; for every plane of the RastPort, you need define data only for planes
    ; that are not entirely zero or one.  As you define your Imagery, you will
    ; often find that most of the planes ARE just as color selectors.  For
    ; instance, if you're designing a two-color Gadget to use colors two and
    ; three, and the Gadget will reside in a five-plane display, plane zero
    ; of your imagery would be all ones, bit plane one would have data that
    ; describes the imagery, and bit planes two through four would be
    ; all zeroes.  Using these flags allows you to avoid wasting all that
    ; memory in this way:
    ; first, you specify which planes you want your data to appear
    ; in using the PlanePick variable.  For each bit set in the variable, the
    ; next "plane" of your image data is blitted to the display.  For each bit
    ; clear in this variable, the corresponding bit in PlaneOnOff is examined.
    ; If that bit is clear, a "plane" of zeroes will be used.  If the bit is
    ; set, ones will go out instead.  So, for our example:
    ;   Gadget.PlanePick = 0x02;
    ;   Gadget.PlaneOnOff = 0x01;
    ; Note that this also allows for generic Gadgets, like the System Gadgets,
    ; which will work in any number of bit planes
    ; Note also that if you want an Image that is only a filled rectangle,
    ; you can get this by setting PlanePick to zero (pick no planes of data)
    ; and set PlaneOnOff to describe the pen color of the rectangle.
    ;
    ; NOTE:  Intuition relies on PlanePick to know how many planes
    ; of data are found in ImageData.  There should be no more
    ; '1'-bits in PlanePick than there are planes in ImageData.
    BYTE ig_PlanePick
    BYTE ig_PlaneOnOff

    ; if the NextImage variable is not NULL, Intuition presumes that
    ; it points to another Image structure with another Image to be
    ; rendered
    APTR ig_NextImage


    LABEL ig_SIZEOF




; ========================================================================
; === IntuiMessage =======================================================
```

```
; ========================================================================
 STRUCTURE IntuiMessage,0

    STRUCT im_ExecMessage,MN_SIZE

    ; the Class bits correspond directly with the IDCMP Flags, except for the
    ; special bit IDCMP_LONELYMESSAGE (defined below)
    LONG im_Class

    ; the Code field is for special values like MENU number
    WORD im_Code

    ; the Qualifier field is a copy of the current InputEvent's Qualifier
    WORD im_Qualifier

    ; IAddress contains particular addresses for Intuition functions, like
    ; the pointer to the Gadget or the Screen
    APTR im_IAddress

    ; when getting mouse movement reports, any event you get will have the
    ; the mouse coordinates in these variables.  the coordinates are relative
    ; to the upper-left corner of your Window (WFLG_GIMMEZEROZERO
    ; notwithstanding)
    ; If the DELTAMOVE IDCMP flag is set, these values will be deltas from
    ; the last reported position.
    WORD im_MouseX
    WORD im_MouseY

    ; the time values are copies of the current system clock time.  Micros
    ; are in units of microseconds, Seconds in seconds.
    LONG im_Seconds
    LONG im_Micros

    ; the IDCMPWindow variable will always have the address of the Window of
    ; this IDCMP
    APTR im_IDCMPWindow

    ; system-use variable
    APTR im_SpecialLink

    LABEL  im_SIZEOF



; --- IDCMP Classes ------------------------------------------------------
; Please refer to the Autodoc for OpenWindow() and to the Rom Kernel
; Manual for full details on the IDCMP classes.

IDCMP_SIZEVERIFY        EQU     $00000001
IDCMP_NEWSIZE           EQU     $00000002
IDCMP_REFRESHWINDOW     EQU     $00000004
IDCMP_MOUSEBUTTONS      EQU     $00000008
IDCMP_MOUSEMOVE         EQU     $00000010
IDCMP_GADGETDOWN        EQU     $00000020
IDCMP_GADGETUP          EQU     $00000040
IDCMP_REQSET            EQU     $00000080
IDCMP_MENUPICK          EQU     $00000100
IDCMP_CLOSEWINDOW       EQU     $00000200
IDCMP_RAWKEY            EQU     $00000400
IDCMP_REQVERIFY         EQU     $00000800
IDCMP_REQCLEAR          EQU     $00001000
IDCMP_MENUVERIFY        EQU     $00002000
IDCMP_NEWPREFS          EQU     $00004000
IDCMP_DISKINSERTED      EQU     $00008000
IDCMP_DISKREMOVED       EQU     $00010000
IDCMP_WBENCHMESSAGE     EQU     $00020000       ; System use only
```

```
IDCMP_ACTIVEWINDOW      EQU     $00040000
IDCMP_INACTIVEWINDOW    EQU     $00080000
IDCMP_DELTAMOVE         EQU     $00100000
IDCMP_VANILLAKEY        EQU     $00200000
IDCMP_INTUITICKS        EQU     $00400000
;  for notifications from "boopsi" gadgets:
IDCMP_IDCMPUPDATE       EQU     $00800000       ; new for V36
; for getting help key report during menu session:
IDCMP_MENUHELP          EQU     $01000000       ; new for V36
; for notification of any move/size/zoom/change window:
IDCMP_CHANGEWINDOW      EQU     $02000000       ; new for V36
; NOTEZ-BIEN:                   $80000000 is reserved for internal use by IDCMP

; the IDCMP Flags do not use this special bit, which is cleared when
; Intuition sends its special message to the Task, and set when Intuition
; gets its Message back from the Task.  Therefore, I can check here to
; find out fast whether or not this Message is available for me to send
IDCMP_LONELYMESSAGE     EQU     $80000000



; --- IDCMP Codes ---------------------------------------------------
; This group of codes is for the IDCMP_MENUVERIFY function
MENUHOT         EQU     $0001   ; IntuiWants verification or MENUCANCEL
MENUCANCEL      EQU     $0002   ; HOT Reply of this cancels Menu operation
MENUWAITING     EQU     $0003   ; Intuition simply wants a ReplyMsg() ASAP

; These are internal tokens to represent state of verification attempts
; shown here as a clue.
OKOK            EQU     MENUHOT         ; guy didn't care
OKABORT         EQU     $0004           ; window rendered question moot
OKCANCEL        EQU     MENUCANCEL      ; window sent cancel reply

; This group of codes is for the IDCMP_WBENCHMESSAGE messages
WBENCHOPEN      EQU $0001
WBENCHCLOSE     EQU $0002

; A data structure common in V36 Intuition processing

 STRUCTURE      IBox,0
    WORD        ibox_Left
    WORD        ibox_Top
    WORD        ibox_Width
    WORD        ibox_Height
 LABEL  ibox_SIZEOF


; =====================================================================
; === Window ==========================================================
; =====================================================================
 STRUCTURE Window,0

    APTR wd_NextWindow          ; for the linked list of a Screen

    WORD wd_LeftEdge            ; screen dimensions
    WORD wd_TopEdge             ; screen dimensions
    WORD wd_Width               ; screen dimensions
    WORD wd_Height              ; screen dimensions

    WORD wd_MouseY              ; relative top top-left corner
    WORD wd_MouseX              ; relative top top-left corner

    WORD wd_MinWidth            ; minimum sizes
    WORD wd_MinHeight           ; minimum sizes
    WORD wd_MaxWidth            ; maximum sizes
    WORD wd_MaxHeight           ; maximum sizes
```

```

    LONG wd_Flags               ; see below for definitions

    APTR wd_MenuStrip           ; first in a list of menu headers

    APTR wd_Title               ; title text for the Window

    APTR wd_FirstRequest        ; first in linked list of active Requesters
    APTR wd_DMRequest           ; the double-menu Requester
    WORD wd_ReqCount            ; number of Requesters blocking this Window
    APTR wd_WScreen             ; this Window's Screen
    APTR wd_RPort               ; this Window's very own RastPort

    ; the border variables describe the window border.  If you specify
    ; WFLG_GIMMEZEROZERO when you open the window, then the upper-left of the
    ; ClipRect for this window will be upper-left of the BitMap (with correct
    ; offsets when in SuperBitMap mode; you MUST select WFLG_GIMMEZEROZERO
    ; when using SuperBitMap).  If you don't specify ZeroZero, then you save
    ; memory (no allocation of RastPort, Layer, ClipRect and associated
    ; Bitmaps), but you also must offset all your writes by BorderTop,
    ; BorderLeft and do your own mini-clipping to prevent writing over the
    ; system gadgets
    BYTE wd_BorderLeft
    BYTE wd_BorderTop
    BYTE wd_BorderRight
    BYTE wd_BorderBottom
    APTR wd_BorderRPort

    ; You supply a linked-list of gadget that you want for your Window.
    ; This list DOES NOT include system Gadgets.  You get the standard
    ; window system Gadgets by setting flag-bits in the variable Flags (see
    ; the bit definitions below)
    APTR wd_FirstGadget

    ; these are for opening/closing the windows
    APTR wd_Parent
    APTR wd_Descendant

    ; sprite data information for your own Pointer
    ; set these AFTER you Open the Window by calling SetPointer()
    APTR wd_Pointer
    BYTE wd_PtrHeight
    BYTE wd_PtrWidth
    BYTE wd_XOffset
    BYTE wd_YOffset

    ; the IDCMP Flags and User's and Intuition's Message Ports
    ULONG wd_IDCMPFlags
    APTR wd_UserPort
    APTR wd_WindowPort
    APTR wd_MessageKey

    BYTE wd_DetailPen
    BYTE wd_BlockPen

    ; the CheckMark is a pointer to the imagery that will be used when
    ; rendering MenuItems of this Window that want to be checkmarked
    ; if this is equal to NULL, you'll get the default imagery
    APTR wd_CheckMark

    ; if non-null, Screen title when Window is active
    APTR wd_ScreenTitle

    ; These variables have the mouse coordinates relative to the
    ; inner-Window of WFLG_GIMMEZEROZERO Windows.  This is compared with the
    ; MouseX and MouseY variables, which contain the mouse coordinates
```

```
	; relative to the upper-left corner of the Window, WFLG_GIMMEZEROZERO
	; notwithstanding
	WORD wd_GZZMouseX
	WORD wd_GZZMouseY
	; these variables contain the width and height of the inner-Window of
	; WFLG_GIMMEZEROZERO Windows
	WORD wd_GZZWidth
	WORD wd_GZZHeight

	APTR wd_ExtData

	; general-purpose pointer to User data extension
	APTR wd_UserData
	APTR wd_WLayer	; stash of Window.RPort->Layer

	; NEW 1.2: need to keep track of the font that OpenWindow opened,
	; in case user SetFont's into RastPort
	APTR  wd_IFont

	; (V36) another flag word (the Flags field is used up).
	; At present, all flag values are system private.
	; Until further notice, you may not change nor use this field.
	ULONG wd_MoreFlags

	; ----- subsequent fields are INTUITION PRIVATE ---

	LABEL wd_Size
	LABEL wd_SIZEOF	; you should never use this: only Intuition allocates

; --- Flags requested at OpenWindow() time by the application -------------
WFLG_SIZEGADGET		EQU $0001	; include sizing system-gadget?
WFLG_DRAGBAR		EQU $0002	; include dragging system-gadget?
WFLG_DEPTHGADGET	EQU $0004	; include depth arrangement gadget?
WFLG_CLOSEGADGET	EQU $0008	; include close-box system-gadget?

WFLG_SIZEBRIGHT		EQU $0010	; size gadget uses right border
WFLG_SIZEBBOTTOM	EQU $0020	; size gadget uses bottom border

; --- refresh modes ----------------------------------------------------------
; combinations of the WFLG_REFRESHBITS select the refresh type
WFLG_REFRESHBITS	EQU $00C0
WFLG_SMART_REFRESH	EQU $0000
WFLG_SIMPLE_REFRESH	EQU $0040
WFLG_SUPER_BITMAP	EQU $0080
WFLG_OTHER_REFRESH	EQU $00C0

WFLG_BACKDROP		EQU $0100	; this is a backdrop window

WFLG_REPORTMOUSE	EQU $0200	; set this to hear every mouse move

WFLG_GIMMEZEROZERO	EQU $0400	; make extra border stuff

WFLG_BORDERLESS		EQU $0800	; set this to get a Window sans border

WFLG_ACTIVATE		EQU $1000	; when Window opens, it's the Active
					; one

; FLAGS SET BY INTUITION
WFLG_WINDOWACTIVE	EQU $2000	; this window is the active one
WFLG_INREQUEST		EQU $4000	; this window is in request mode
WFLG_MENUSTATE		EQU $8000	; this Window is active with its
					; Menus on

; --- Other User Flags -------------------------------------------------------
WFLG_RMBTRAP		EQU $00010000	; Catch RMB events for your own
WFLG_NOCAREREFRESH	EQU $00020000	; not to be bothered with REFRESH
```

```

; --- Other Intuition Flags ---------------------------------------------
WFLG_WINDOWREFRESH	EQU $01000000	; Window is currently refreshing
WFLG_WBENCHWINDOW	EQU $02000000	; WorkBench Window
WFLG_WINDOWTICKED	EQU $04000000	; only one timer tick at a time

SUPER_UNUSED	EQU $FCFC0000	;bits of Flag unused yet

; - V36 new Flags which the programmer may specify in NewScreen.Flags
WFLG_NW_EXTENDED	EQU $00040000	; extension data provided
					; see ExtNewWindow structure

; --- V36 Flags to be set only by Intuition -------------------------
WFLG_VISITOR		EQU $08000000	; visitor window (see autodoc for OpenWi
  ndow)
WFLG_ZOOMED		EQU $10000000	; identifies "zoom state"
WFLG_HASZOOM		EQU $20000000	; window has a zoom gadget

; --- Other Window Values -------------------------------------------
DEFAULTMOUSEQUEUE	EQU 5	; no more mouse messages


; --- see struct IntuiMessage for the IDCMP Flag definitions -----------------


; ======================================================================
; === NewWindow ========================================================
; ======================================================================
; NOTE: to use the new features of V36, you may need to use the
; ExtNewWindow structure, below.
 STRUCTURE NewWindow,0

	WORD nw_LeftEdge		; initial Window dimensions
	WORD nw_TopEdge			; initial Window dimensions
	WORD nw_Width			; initial Window dimensions
	WORD nw_Height			; initial Window dimensions

	BYTE nw_DetailPen		; for rendering the detail bits of the Window
	BYTE nw_BlockPen		; for rendering the block-fill bits

	LONG nw_IDCMPFlags		; initial IDCMP state

	LONG nw_Flags			; see the Flag definition under Window

	; You supply a linked-list of Gadgets for your Window.
	; This list DOES NOT include system Gadgets.  You get the standard
	; system Window Gadgets by setting flag-bits in the variable Flags (see
	; the bit definitions under the Window structure definition)
	APTR	nw_FirstGadget

	; the CheckMark is a pointer to the imagery that will be used when
	; rendering MenuItems of this Window that want to be checkmarked
	; if this is equal to NULL, you'll get the default imagery
	APTR nw_CheckMark

	APTR nw_Title			; title text for the Window

	; the Screen pointer is used only if you've defined a CUSTOMSCREEN and
	; want this Window to open in it.  If so, you pass the address of the
	; Custom Screen structure in this variable.  Otherwise, this variable
	; is ignored and doesn't have to be initialized.
	APTR nw_Screen

	; WFLG_SUPER_BITMAP Window?  If so, put the address of your BitMap
	; structure in this variable.  If not, this variable is ignored and
	; doesn't have to be initialized
```

```
    APTR nw_BitMap

    ; the values describe the minimum and maximum sizes of your Windows.
    ; these matter only if you've chosen the WFLG_SIZEGADGET Gadget option,
    ; which means that you want to let the User to change the size of
    ; this Window.  You describe the minimum and maximum sizes that the
    ; Window can grow by setting these variables.  You can initialize
    ; any one these to zero, which will mean that you want to duplicate
    ; the setting for that dimension (if MinWidth == 0, MinWidth will be
    ; set to the opening Width of the Window).
    ; You can change these settings later using SetWindowLimits().
    ; If you haven't asked for a GTYP_SIZING Gadget, you don't have to
    ; initialize any of these variables.
    WORD nw_MinWidth
    WORD nw_MinHeight
    WORD nw_MaxWidth
    WORD nw_MaxHeight

    ; the type variable describes the Screen in which you want this Window to
    ; open.  The type value can either be CUSTOMSCREEN or one of the
    ; system standard Screen Types such as WBENCHSCREEN.  See the
    ; type definitions under the Screen structure
    ; A new possible value for this field is PUBLICSCREEN, which
    ; defines the window as a 'visitor' window.  See below for
    ; additional information provided.
    WORD nw_Type

    LABEL nw_SIZE
    LABEL nw_SIZEOF

; ExtNewWindow -- NewWindow plus extension fields.
; This structure may be extended again, so programs depending on its
; size are incorrect.

 STRUCTURE ExtNewWindow,nw_SIZE

    ; extensions for V36
    ; if the NewWindow Flag WFLG_NW_EXTENDED is set, then
    ; this field is assumed to point to an array (or chain of arrays)
    ; of TagItem structures.  See also ExtNewScreen for another
    ; use of TagItems to pass optional data.
    ;
    ; see below for tag values and the corresponding data

    APTR        enw_Extension   ; pointer to TagItem array
    LABEL enw_SIZEOF

* The TagItem ID's (ti_Tag values) for OpenWindowTagList() follow.
* They are values in a TagItem array passed as extension/replacement
* values for the data in NewWindow.  OpenWindowTagList() can actually
* work well with a NULL NewWindow pointer.

    ENUM TAG_USER+100

    ; these tags simply override NewWindow parameters
    EITEM WA_Left
    EITEM WA_Top
    EITEM WA_Width
    EITEM WA_Height
    EITEM WA_DetailPen
    EITEM WA_BlockPen
    EITEM WA_IDCMP
    EITEM WA_Flags      ; not implemented at present
    EITEM WA_Gadgets
    EITEM WA_Checkmark
    EITEM WA_Title
```

```
    EITEM WA_ScreenTitle        ; means you don't have to call SetWindowTitles
                                ; after you open your window

    EITEM WA_CustomScreen
    EITEM WA_SuperBitMap        ; also implies WFLG_SUPER_BITMAP property
    EITEM WA_MinWidth
    EITEM WA_MinHeight
    EITEM WA_MaxWidth
    EITEM WA_MaxHeight

    ; The following are specifications for new features

    EITEM WA_InnerWidth
    EITEM WA_InnerHeight ; You can specify the dimensions of the interior
                         ; region of your window, independent of what
                         ; the border widths will be.  You probably want
                         ; to also specify WA_AutoAdjust to allow
                         ; Intuition to move your window or even
                         ; shrink it so that it is completely on screen.

    EITEM WA_PubScreenName      ; declares that you want the window to open as
                         ; a visitor on the public screen whose name is
                         ; pointed to by (UBYTE *) ti_Data

    EITEM WA_PubScreen   ; open as a visitor window on the public screen
                         ; whose address is in (struct Screen *) ti_Data.
                         ; To ensure that this screen remains open, you
                         ; should either be the screen's owner, have a
                         ; window open on the screen, or use LockPubScreen().

    EITEM WA_PubScreenFallBack  ; A Boolean, specifies whether a visitor window
                         ; should "fall back" to the default public screen
                         ; (or Workbench) if the named public screen isn't
                         ; available

    EITEM WA_WindowName ; not implemented
    EITEM WA_Colors     ; a ColorSpec array for colors to be set
                        ; when this window is active.  This is not
                        ; implemented, and may not be, since the default
                        ; values to restore would be hard to track.
                        ; We'd like to at least support per-window colors
                        ; for the mouse pointer sprite.

    EITEM WA_Zoom       ; ti_Data points to an array of four WORD's,
                        ; the initial Left/Top/Width/Height values of
                        ; the "alternate" zoom position/dimensions.
                        ; It also specifies that you want a Zoom gadget
                        ; for your window, whether or not you have a
                        ; sizing gadget.

    EITEM WA_MouseQueue ; ti_Data contains initial value for the mouse
                        ; message backlog limit for this window.

    EITEM WA_BackFill   ; unimplemented at present: provides a "backfill
                        ; hook" for your window's layer.

    EITEM WA_RptQueue   ; initial value of repeat key backlog limit

    ; These Boolean tag items are alternatives to the NewWindow.Flags
    ; boolean flags with similar names.

    EITEM WA_SizeGadget
    EITEM WA_DragBar
    EITEM WA_DepthGadget
    EITEM WA_CloseGadget
    EITEM WA_Backdrop
    EITEM WA_ReportMouse
    EITEM WA_NoCareRefresh
    EITEM WA_Borderless
    EITEM WA_Activate
    EITEM WA_RMBTrap
    EITEM WA_WBenchWindow       ; PRIVATE!!
    EITEM WA_SimpleRefresh      ; only specify if TRUE
    EITEM WA_SmartRefresh       ; only specify if TRUE
    EITEM WA_SizeBRight
    EITEM WA_SizeBBottom

    ; New Boolean properties
    EITEM WA_AutoAdjust ; shift or squeeze the window's position and
                        ; dimensions to fit it on screen.

    EITEM WA_GimmeZeroZero      ; equiv. to NewWindow.Flags WFLG_GIMMEZEROZERO

    ; New for V37: WA_MenuHelp (ignored by V36)
    EITEM WA_MenuHelp   ; Enables IDCMP_MENUHELP:  Pressing HELP during menus
                        ; will return IDCMP_MENUHELP IDCMP message.

*** End of Window attribute enumeration ***



        IFND INTUITION_SCREENS_I
        INCLUDE "intuition/screens.i"
        ENDC

        IFND INTUITION_PREFERENCES_I
        INCLUDE "intuition/preferences.i"
        ENDC

; =========================================================================
; === Remember ============================================================
; =========================================================================
; this structure is used for remembering what memory has been allocated to
; date by a given routine, so that a premature abort or systematic exit
; can deallocate memory cleanly, easily, and completely
 STRUCTURE Remember,0

    APTR rm_NextRemember
    LONG rm_RememberSize
    APTR rm_Memory

 LABEL    rm_SIZEOF

 STRUCTURE ColorSpec,0

    WORD cs_ColorIndex  ; -1 terminates an array of ColorSpec
    UWORD cs_Red        ; only six bits recognized in V36
    UWORD cs_Green      ; only six bits recognized in V36
    UWORD cs_Blue       ; only six bits recognized in V36
 LABEL  cs_SIZEOF

* === Easy Requester Specification ======================================== *
* see also autodocs for EasyRequest and BuildEasyRequest
* NOTE: This structure may grow in size in the future

 STRUCTURE EasyStruct,0

    ULONG es_StructSize         ; should be es_SIZEOF
    ULONG es_Flags              ; should be 0 for now
    APTR  es_Title              ; title of requester window
    APTR  es_TextFormat         ; 'printf' style formatting string
    APTR  es_GadgetFormat       ; 'printf' style formatting string
```

```
 LABEL  es_SIZEOF



; ===========================================================================
; === Miscellaneous =========================================================
; ===========================================================================

; = MACROS ================================================================
;#define MENUNUM(n) (n & 0x1F)
;#define ITEMNUM(n) ((n >> 5) & 0x003F)
;#define SUBNUM(n) ((n >> 11) & 0x001F)
;
;#define SHIFTMENU(n) (n & 0x1F)
;#define SHIFTITEM(n) ((n & 0x3F) << 5)
;#define SHIFTSUB(n) ((n & 0x1F) << 11)
;
;#define SRBNUM(n)  (0x08 - (n >> 4))  /* SerRWBits -> read bits per char */
;#define SWBNUM(n)  (0x08 - (n & 0x0F))/* SerRWBits -> write bits per chr */
;#define SSBNUM(n)  (0x01 + (n >> 4))  /* SerStopBuf -> stop bits per chr */
;#define SPARNUM(n) (n >> 4)           /* SerParShk -> parity setting    */
;#define SHAKNUM(n) (n & 0x0F)         /* SerParShk -> handshake mode    */
;
; = MENU STUFF ==============================================================
NOMENU EQU      $001F
NOITEM EQU      $003F
NOSUB  EQU      $001F
MENUNULL EQU    $FFFF


; = =RJ='s peculiarities ====================================================
;#define FOREVER for(;;)
;#define SIGN(x) ( ((x) > 0) - ((x) < 0) )


; these defines are for the COMMSEQ and CHECKIT menu stuff.  If CHECKIT,
; I'll use a generic Width (for all resolutions) for the CheckMark.
; If COMMSEQ, likewise I'll use this generic stuff
CHECKWIDTH      EQU     19
COMMWIDTH       EQU     27
LOWCHECKWIDTH   EQU     13
LOWCOMMWIDTH    EQU     16


; these are the AlertNumber defines.  if you are calling DisplayAlert()
; the AlertNumber you supply must have the ALERT_TYPE bits set to one
; of these patterns
ALERT_TYPE      EQU     $80000000
RECOVERY_ALERT  EQU     $00000000       ; the system can recover from this
DEADEND_ALERT   EQU     $80000000       ; no recovery possible, this is it


; When you're defining IntuiText for the Positive and Negative Gadgets
; created by a call to AutoRequest(), these defines will get you
; reasonable-looking text.  The only field without a define is the IText
; field; you decide what text goes with the Gadget
AUTOFRONTPEN    EQU     0
AUTOBACKPEN     EQU     1
AUTODRAWMODE    EQU     RP_JAM2
AUTOLEFTEDGE    EQU     6
AUTOTOPEDGE     EQU     3
AUTOITEXTFONT   EQU     0
AUTONEXTTEXT    EQU     0
```

```
;* --- RAWMOUSE Codes and Qualifiers (Console OR IDCMP) -------------------
SELECTUP        EQU     (IECODE_LBUTTON+IECODE_UP_PREFIX)
SELECTDOWN      EQU     (IECODE_LBUTTON)
MENUUP          EQU     (IECODE_RBUTTON+IECODE_UP_PREFIX)
MENUDOWN        EQU     (IECODE_RBUTTON)
ALTLEFT         EQU     (IEQUALIFIER_LALT)
ALTRIGHT        EQU     (IEQUALIFIER_RALT)
AMIGALEFT       EQU     (IEQUALIFIER_LCOMMAND)
AMIGARIGHT      EQU     (IEQUALIFIER_RCOMMAND)
AMIGAKEYS       EQU     (AMIGALEFT+AMIGARIGHT)

CURSORUP        EQU     $4C
CURSORLEFT      EQU     $4F
CURSORRIGHT     EQU     $4E
CURSORDOWN      EQU     $4D
KEYCODE_Q       EQU     $10
KEYCODE_X       EQU     $32
KEYCODE_N       EQU     $36
KEYCODE_M       EQU     $37
KEYCODE_V       EQU     $34
KEYCODE_B       EQU     $35
KEYCODE_LESS    EQU     $38
KEYCODE_GREATER EQU     $39

        IFND    INTUITION_INTUITIONBASE_I
        INCLUDE "intuition/intuitionbase.i"
        ENDC

* Include obsolete identifiers:
        IFND    INTUITION_IOBSOLETE_I
        INCLUDE "intuition/iobsolete.i"
        ENDC

        ENDC
```

intuition/intuitionbase.i

```
        IFND  INTUITION_INTUITIONBASE_I
INTUITION_INTUITIONBASE_I SET  1
**
** $Filename: intuition/intuitionbase.i $
** $Release: 2.04 $
** $Revision: 36.5 $
** $Date: 90/07/12 $
**
** The public part of  IntuitionBase structure and supporting structures
**
**  (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**

        IFND EXEC_TYPES_I
        INCLUDE "exec/types.i"
        ENDC

        IFND EXEC_LIBRARIES_I
        INCLUDE "exec/libraries.i"
        ENDC

        IFND    GRAPHICS_VIEW_I
        INCLUDE "graphics/view.i"
        ENDC

* Be sure to protect yourself against someone modifying these data as
* you look at them.  This is done by calling:
*
* lock = LockIBase(0), which returns a ULONG.  When done call
*  D0              D0
* UnlockIBase(lock) where lock is what LockIBase() returned.
*               A0
* NOTE: these library functions are simply stubs now, but should be called
* to be compatible with future releases.

* ===================================================================== *
* === IntuitionBase =================================================== *
* ===================================================================== *
 STRUCTURE IntuitionBase,0

        STRUCT  ib_LibNode,LIB_SIZE
        STRUCT  ib_ViewLord,v_SIZEOF
        APTR    ib_ActiveWindow
        APTR    ib_ActiveScreen

* the FirstScreen variable points to the frontmost Screen.  Screens are
* then maintained in a front to back order using Screen.NextScreen

        APTR    ib_FirstScreen
        ULONG   ib_Flags        ; private meaning
        WORD    ib_MouseY       ; these are supposed to be backwards,
        WORD    ib_MouseX       ;  but weren't, recently

        ULONG   ib_Seconds
        ULONG   ib_Micros

* there is not 'sizeof' here because...
*
*
        ENDC


	IFND INTUITION_IOBSOLETE_I
INTUITION_IOBSOLETE_I SET 1
**
**  $Filename: intuition/iobsolete.i $
**  $Release: 2.04 $
**  $Revision: 36.2 $
**  $Date: 91/02/05 $
**
**  Obsolete identifiers for Intuition.  Use the new ones instead!
**
**  (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**	    All Rights Reserved
**


* This file contains:
*
* 1.  The traditional identifiers for gadget Flags, Activation, and Type,
* and for window Flags and IDCMP classes.  They are defined in terms
* of their new versions, which serve to prevent confusion between
* similar-sounding but different identifiers (like IDCMP_WINDOWACTIVE
* and WFLG_ACTIVATE).
*
* 2.  Some tag names and constants whose labels were adjusted after V36.
*
* By default, 1 and 2 are enabled.
*
* Set INTUI_V36_NAMES_ONLY to exclude the traditional identifiers and
* the original V36 names of some identifiers.
*


	IFND INTUITION_INTUITION_I
	INCLUDE "intuition/intuition.i"
	ENDC

	IFND INTUITION_SCREENS_I
	INCLUDE "intuition/screens.i"
	ENDC

	IFND INTUITION_GADGETCLASS_I
	INCLUDE "intuition/gadgetclass.i"
	ENDC

	IFND INTUITION_IMAGECLASS_I
	INCLUDE "intuition/imageclass.i"
	ENDC

* Set INTUI_V36_NAMES_ONLY to remove these older names

	IFND INTUI_V36_NAMES_ONLY


* V34-style Gadget->Flags names:

GADGHIGHBITS	equ	GFLG_GADGHIGHBITS
GADGHCOMP	equ	GFLG_GADGHCOMP
GADGHBOX	equ	GFLG_GADGHBOX
GADGHIMAGE	equ	GFLG_GADGHIMAGE
GADGHNONE	equ	GFLG_GADGHNONE
GADGIMAGE	equ	GFLG_GADGIMAGE
GRELBOTTOM	equ	GFLG_RELBOTTOM
GRELRIGHT	equ	GFLG_RELRIGHT
GRELWIDTH	equ	GFLG_RELWIDTH
GRELHEIGHT	equ	GFLG_RELHEIGHT
SELECTED	equ	GFLG_SELECTED
```

```
GADGDISABLED	equ	GFLG_DISABLED
LABELMASK	equ	GFLG_LABELMASK
LABELITEXT	equ	GFLG_LABELITEXT
LABELSTRING	equ	GFLG_LABELSTRING
LABELIMAGE	equ	GFLG_LABELIMAGE


* V34-style Gadget->Activation flag names:

RELVERIFY	equ	GACT_RELVERIFY
GADGIMMEDIATE	equ	GACT_IMMEDIATE
ENDGADGET	equ	GACT_ENDGADGET
FOLLOWMOUSE	equ	GACT_FOLLOWMOUSE
RIGHTBORDER	equ	GACT_RIGHTBORDER
LEFTBORDER	equ	GACT_LEFTBORDER
TOPBORDER	equ	GACT_TOPBORDER
BOTTOMBORDER	equ	GACT_BOTTOMBORDER
BORDERSNIFF	equ	GACT_BORDERSNIFF
TOGGLESELECT	equ	GACT_TOGGLESELECT
BOOLEXTEND	equ	GACT_BOOLEXTEND
STRINGLEFT	equ	GACT_STRINGLEFT
STRINGCENTER	equ	GACT_STRINGCENTER
STRINGRIGHT	equ	GACT_STRINGRIGHT
LONGINT		equ	GACT_LONGINT
ALTKEYMAP	equ	GACT_ALTKEYMAP
STRINGEXTEND	equ	GACT_STRINGEXTEND
ACTIVEGADGET	equ	GACT_ACTIVEGADGET


* V34-style Gadget->Type names:

GADGETTYPE	equ	GTYP_GADGETTYPE
SYSGADGET	equ	GTYP_SYSGADGET
SCRGADGET	equ	GTYP_SCRGADGET
GZZGADGET	equ	GTYP_GZZGADGET
REQGADGET	equ	GTYP_REQGADGET
SIZING		equ	GTYP_SIZING
WDRAGGING	equ	GTYP_WDRAGGING
SDRAGGING	equ	GTYP_SDRAGGING
WUPFRONT	equ	GTYP_WUPFRONT
SUPFRONT	equ	GTYP_SUPFRONT
WDOWNBACK	equ	GTYP_WDOWNBACK
SDOWNBACK	equ	GTYP_SDOWNBACK
CLOSE		equ	GTYP_CLOSE
BOOLGADGET	equ	GTYP_BOOLGADGET
GADGET0002	equ	GTYP_GADGET0002
PROPGADGET	equ	GTYP_PROPGADGET
STRGADGET	equ	GTYP_STRGADGET
CUSTOMGADGET	equ	GTYP_CUSTOMGADGET
GTYPEMASK	equ	GTYP_GTYPEMASK


* V34-style IDCMP class names:

SIZEVERIFY	equ	IDCMP_SIZEVERIFY
NEWSIZE		equ	IDCMP_NEWSIZE
REFRESHWINDOW	equ	IDCMP_REFRESHWINDOW
MOUSEBUTTONS	equ	IDCMP_MOUSEBUTTONS
MOUSEMOVE	equ	IDCMP_MOUSEMOVE
GADGETDOWN	equ	IDCMP_GADGETDOWN
GADGETUP	equ	IDCMP_GADGETUP
REQSET		equ	IDCMP_REQSET
MENUPICK	equ	IDCMP_MENUPICK
CLOSEWINDOW	equ	IDCMP_CLOSEWINDOW
RAWKEY		equ	IDCMP_RAWKEY
REQVERIFY	equ	IDCMP_REQVERIFY
```

```
REQCLEAR        equ     IDCMP_REQCLEAR
MENUVERIFY      equ     IDCMP_MENUVERIFY
NEWPREFS        equ     IDCMP_NEWPREFS
DISKINSERTED    equ     IDCMP_DISKINSERTED
DISKREMOVED     equ     IDCMP_DISKREMOVED
WBENCHMESSAGE   equ     IDCMP_WBENCHMESSAGE
ACTIVEWINDOW    equ     IDCMP_ACTIVEWINDOW
INACTIVEWINDOW  equ     IDCMP_INACTIVEWINDOW
DELTAMOVE       equ     IDCMP_DELTAMOVE
VANILLAKEY      equ     IDCMP_VANILLAKEY
INTUITICKS      equ     IDCMP_INTUITICKS
IDCMPUPDATE     equ     IDCMP_IDCMPUPDATE
MENUHELP        equ     IDCMP_MENUHELP
CHANGEWINDOW    equ     IDCMP_CHANGEWINDOW
LONELYMESSAGE   equ     IDCMP_LONELYMESSAGE


* V34-style Window->Flags names:

WINDOWSIZING    equ     WFLG_SIZEGADGET
WINDOWDRAG      equ     WFLG_DRAGBAR
WINDOWDEPTH     equ     WFLG_DEPTHGADGET
WINDOWCLOSE     equ     WFLG_CLOSEGADGET
SIZEBRIGHT      equ     WFLG_SIZEBRIGHT
SIZEBBOTTOM     equ     WFLG_SIZEBBOTTOM
REFRESHBITS     equ     WFLG_REFRESHBITS
SMART_REFRESH   equ     WFLG_SMART_REFRESH
SIMPLE_REFRESH  equ     WFLG_SIMPLE_REFRESH
SUPER_BITMAP    equ     WFLG_SUPER_BITMAP
OTHER_REFRESH   equ     WFLG_OTHER_REFRESH
BACKDROP        equ     WFLG_BACKDROP
REPORTMOUSE     equ     WFLG_REPORTMOUSE
GIMMEZEROZERO   equ     WFLG_GIMMEZEROZERO
BORDERLESS      equ     WFLG_BORDERLESS
ACTIVATE        equ     WFLG_ACTIVATE
WINDOWACTIVE    equ     WFLG_WINDOWACTIVE
INREQUEST       equ     WFLG_INREQUEST
MENUSTATE       equ     WFLG_MENUSTATE
RMBTRAP         equ     WFLG_RMBTRAP
NOCAREREFRESH   equ     WFLG_NOCAREREFRESH
WINDOWREFRESH   equ     WFLG_WINDOWREFRESH
WBENCHWINDOW    equ     WFLG_WBENCHWINDOW
WINDOWTICKED    equ     WFLG_WINDOWTICKED
NW_EXTENDED     equ     WFLG_NW_EXTENDED
VISITOR         equ     WFLG_VISITOR
ZOOMED          equ     WFLG_ZOOMED
HASZOOM         equ     WFLG_HASZOOM


* These are the obsolete tag names for general gadgets, proportional gadgets,
* and string gadgets.  Use the mixed-case equivalents from gadgetclass.h
* instead.
*

GA_LEFT                 equ     GA_Left
GA_RELRIGHT             equ     GA_RelRight
GA_TOP                  equ     GA_Top
GA_RELBOTTOM            equ     GA_RelBottom
GA_WIDTH                equ     GA_Width
GA_RELWIDTH             equ     GA_RelWidth
GA_HEIGHT               equ     GA_Height
GA_RELHEIGHT            equ     GA_RelHeight
GA_TEXT                 equ     GA_Text
GA_IMAGE                equ     GA_Image
GA_BORDER               equ     GA_Border
GA_SELECTRENDER         equ     GA_SelectRender
```

```
GA_HIGHLIGHT            equ     GA_Highlight
GA_DISABLED             equ     GA_Disabled
GA_GZZGADGET            equ     GA_GZZGadget
GA_USERDATA             equ     GA_UserData
GA_SPECIALINFO          equ     GA_SpecialInfo
GA_SELECTED             equ     GA_Selected
GA_ENDGADGET            equ     GA_EndGadget
GA_IMMEDIATE            equ     GA_Immediate
GA_RELVERIFY            equ     GA_RelVerify
GA_FOLLOWMOUSE          equ     GA_FollowMouse
GA_RIGHTBORDER          equ     GA_RightBorder
GA_LEFTBORDER           equ     GA_LeftBorder
GA_TOPBORDER            equ     GA_TopBorder
GA_BOTTOMBORDER         equ     GA_BottomBorder
GA_TOGGLESELECT         equ     GA_ToggleSelect
GA_SYSGADGET            equ     GA_SysGadget
GA_SYSGTYPE             equ     GA_SysGType
GA_PREVIOUS             equ     GA_Previous
GA_NEXT                 equ     GA_Next
GA_DRAWINFO             equ     GA_DrawInfo
GA_INTUITEXT            equ     GA_IntuiText
GA_LABELIMAGE           equ     GA_LabelImage

PGA_FREEDOM             equ     PGA_Freedom
PGA_BORDERLESS          equ     PGA_Borderless
PGA_HORIZPOT            equ     PGA_HorizPot
PGA_HORIZBODY           equ     PGA_HorizBody
PGA_VERTPOT             equ     PGA_VertPot
PGA_VERTBODY            equ     PGA_VertBody
PGA_TOTAL               equ     PGA_Total
PGA_VISIBLE             equ     PGA_Visible
PGA_TOP                 equ     PGA_Top

LAYOUTA_LAYOUTOBJ       equ     LAYOUTA_LayoutObj
LAYOUTA_SPACING         equ     LAYOUTA_Spacing
LAYOUTA_ORIENTATION     equ     LAYOUTA_Orientation


* These are the obsolete tag names for image attributes.
* Use the mixed-case equivalents from imageclass.h instead.
*

IA_LEFT                 equ     IA_Left
IA_TOP                  equ     IA_Top
IA_WIDTH                equ     IA_Width
IA_HEIGHT               equ     IA_Height
IA_FGPEN                equ     IA_FGPen
IA_BGPEN                equ     IA_BGPen
IA_DATA                 equ     IA_Data
IA_LINEWIDTH            equ     IA_LineWidth
IA_PENS                 equ     IA_Pens
IA_RESOLUTION           equ     IA_Resolution
IA_APATTERN             equ     IA_APattern
IA_APATSIZE             equ     IA_APatSize
IA_MODE                 equ     IA_Mode
IA_FONT                 equ     IA_Font
IA_OUTLINE              equ     IA_Outline
IA_RECESSED             equ     IA_Recessed
IA_DOUBLEEMBOSS         equ     IA_DoubleEmboss
IA_EDGESONLY            equ     IA_EdgesOnly
IA_SHADOWPEN            equ     IA_ShadowPen
IA_HIGHLIGHTPEN         equ     IA_HighlightPen


* These are the obsolete identifiers for the various DrawInfo pens.
* Use the uppercase versions in screens.h instead.
```

intuition/iobsolete.i

```
*

detailPen        equ     DETAILPEN
blockPen         equ     BLOCKPEN
textPen          equ     TEXTPEN
shinePen         equ     SHINEPEN
shadowPen        equ     SHADOWPEN
hifillPen        equ     FILLPEN
hifilltextPen    equ     FILLTEXTPEN
backgroundPen    equ     BACKGROUNDPEN
hilighttextPen   equ     HIGHLIGHTTEXTPEN
numDrIPens       equ     NUMDRIPENS


        ENDC    * !INTUI_V36_NAMES_ONLY

        ENDC    * INTUITION_IOBSOLETE_I
```

intuition/preferences.i

```
        IFND  INTUITION_PREFERENCES_I
INTUITION_PREFERENCES_I SET  1
**
**      $Filename: intuition/preferences.i $
**      $Release: 2.04 $
**      $Revision: 36.5 $
**      $Date: 90/11/02 $
**
**      Structure definition for old-style preferences
**
**      (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**

        IFND EXEC_TYPES_I
        INCLUDE "exec/types.i"
        ENDC

        IFND    DEVICES_TIMER_I
        INCLUDE "devices/timer.i"
        ENDC

; ======================================================================
; === Preferences ======================================================
; ======================================================================

; these are the definitions for the printer configurations
FILENAME_SIZE EQU       30      ; Filename size

POINTERSIZE     EQU     (1+16+1)*2      ; Size of Pointer data buffer

; These defines are for the default font size.  These actually describe the
; height of the defaults fonts.  The default font type is the topaz
; font, which is a fixed width font that can be used in either
; eighty-column or sixty-column mode.  The Preferences structure reflects
; which is currently selected by the value found in the variable FontSize,
; which may have either of the values defined below.  These values actually
; are used to select the height of the default font.  By changing the
; height, the resolution of the font changes as well.
TOPAZ_EIGHTY    EQU     8
TOPAZ_SIXTY     EQU     9

; ----------------------------------------------------------------------
 STRUCTURE Preferences,0

    ; the default font height
    BYTE pf_FontHeight          ; height for system default font

    ; constant describing what's hooked up to the port
    BYTE pf_PrinterPort ; printer port connection

    ; the baud rate of the port
    WORD pf_BaudRate            ; baud rate for the serial port

    ; various timing rates
    STRUCT pf_KeyRptSpeed,TV_SIZE ; repeat speed for keyboard
    STRUCT pf_KeyRptDelay,TV_SIZE ; Delay before keys repeat
    STRUCT pf_DoubleClick,TV_SIZE ; Interval allowed between clicks

    ; Intuition Pointer data
    STRUCT pf_PointerMatrix,POINTERSIZE*2 ; Definition of pointer sprite
    BYTE pf_XOffset             ; X-Offset for active 'bit'
    BYTE pf_YOffset             ; Y-Offset for active 'bit'
    WORD pf_color17             ;********************************
    WORD pf_color18             ; Colours for sprite pointer
    WORD pf_color19             ;********************************
    WORD pf_PointerTicks            ; Sensitivity of the pointer

    ; Workbench Screen colors
    WORD pf_color0                  ;********************************
    WORD pf_color1                  ;  Standard default colours
    WORD pf_color2                  ;   Used in the Workbench
    WORD pf_color3                  ;********************************

    ; positioning data for the Intuition View
    BYTE pf_ViewXOffset             ; Offset for top lefthand corner
    BYTE pf_ViewYOffset             ; X and Y dimensions
    WORD pf_ViewInitX               ; View initial offsets at startup
    WORD pf_ViewInitY               ; View initial offsets at startup

    BOOL EnableCLI                  ; CLI availability switch

    ; printer configurations
    WORD pf_PrinterType             ; printer type
    STRUCT pf_PrinterFilename,FILENAME_SIZE ; file for printer

    ; print format and quality configurations
    WORD pf_PrintPitch              ; print pitch
    WORD pf_PrintQuality            ; print quality
    WORD pf_PrintSpacing            ; number of lines per inch
    WORD pf_PrintLeftMargin         ; left margin in characters
    WORD pf_PrintRightMargin        ; right margin in characters
    WORD pf_PrintImage              ; positive or negative
    WORD pf_PrintAspect             ; horizontal or vertical
    WORD pf_PrintShade              ; b&w, half-tone, or color
    WORD pf_PrintThreshold          ; darkness ctrl for b/w dumps


    ; print paper description
    WORD pf_PaperSize               ; paper size
    WORD pf_PaperLength             ; paper length in lines
    WORD pf_PaperType               ; continuous or single sheet

    ; Serial device settings: These are six nibble-fields in three bytes
    ; (these look a little strange so the defaults will map out to zero)
    BYTE pf_SerRWBits               ; upper nibble = (8-number of read bits)
                                        ; lower nibble = (8-number of write bits)
    BYTE pf_SerStopBuf              ; upper nibble = (number of stop bits - 1)
                                    ; lower nibble = (table value for BufSize)
    BYTE pf_SerParShk               ; upper nibble = (value for Parity setting)
                                    ; lower nibble = (value for Handshake mode)

    BYTE pf_LaceWB                  ; if workbench is to be interlaced

    STRUCT pf_WorkName,FILENAME_SIZE ; temp file for printer

    BYTE    pf_RowSizeChange    ;
    BYTE    pf_ColumnSizeChange ;

    UWORD   pf_PrintFlags       ; user preference flags
    WORD    pf_PrintMaxWidth    ; max width of printed picture in 10ths/inch
    UWORD   pf_PrintMaxHeight   ; max height of printed picture in 10ths/inch
    UBYTE   pf_PrintDensity     ; print density
    UBYTE   pf_PrintXOffset     ; offset of printed picture in 10ths/inch

    UWORD   pf_wb_Width         ; override default workbench width
    UWORD   pf_wb_Height        ; override default workbench height
    UBYTE   pf_wb_Depth         ; override default workbench depth

    UBYTE   pf_ext_size         ; extension information -- do not touch!
                                ; extension size in blocks of 64 bytes

```

```
    LABEL pf_SIZEOF


; === Preferences definitions ==========================================

; Workbench Interlace (use one bit)
LACEWB          EQU     $01

; PrinterPort
PARALLEL_PRINTER EQU    $00
SERIAL_PRINTER  EQU     $01

; BaudRate
BAUD_110        EQU     $00
BAUD_300        EQU     $01
BAUD_1200       EQU     $02
BAUD_2400       EQU     $03
BAUD_4800       EQU     $04
BAUD_9600       EQU     $05
BAUD_19200      EQU     $06
BAUD_MIDI       EQU     $07

; PaperType
FANFOLD         EQU     $00
SINGLE          EQU     $80

; PrintPitch
PICA            EQU     $000
ELITE           EQU     $400
FINE            EQU     $800

; PrintQuality
DRAFT           EQU     $000
LETTER          EQU     $100

; PrintSpacing
SIX_LPI         EQU     $000
EIGHT_LPI       EQU     $200

; Print Image
IMAGE_POSITIVE  EQU     $00
IMAGE_NEGATIVE  EQU     $01

; PrintAspect
ASPECT_HORIZ    EQU     $00
ASPECT_VERT     EQU     $01

; PrintShade
SHADE_BW        EQU     $00
SHADE_GREYSCALE EQU     $01
SHADE_COLOR     EQU     $02

; PaperSize
US_LETTER       EQU     $00
US_LEGAL        EQU     $10
N_TRACTOR       EQU     $20
W_TRACTOR       EQU     $30
CUSTOM          EQU     $40

; PrinterType
CUSTOM_NAME     EQU     $00
ALPHA_P_101     EQU     $01
BROTHER_15XL    EQU     $02
CBM_MPS1000     EQU     $03
DIAB_630        EQU     $04
DIAB_ADV_D25    EQU     $05
```

intuition/preferences.i Page 4

```
DIAB_C_150      EQU     $06
EPSON           EQU     $07
EPSON_JX_80     EQU     $08
OKIMATE_20      EQU     $09
QUME_LP_20      EQU     $0A
; new printer entries, 3 October 1985
HP_LASERJET     EQU     $0B
HP_LASERJET_PLUS EQU    $0C


; Serial Input Buffer Sizes
SBUF_512        EQU     $00
SBUF_1024       EQU     $01
SBUF_2048       EQU     $02
SBUF_4096       EQU     $03
SBUF_8000       EQU     $04
SBUF_16000      EQU     $05

; Serial Bit Masks
SREAD_BITS      EQU     $F0     ; pf_SerRWBits
SWRITE_BITS     EQU     $0F

SSTOP_BITS      EQU     $F0     ; pf_SerStopBuf
SBUFSIZE_BITS   EQU     $0F

SPARITY_BITS    EQU     $F0     ; pf_SerParShk
SHSHAKE_BITS    EQU     $0F

; Serial Parity (high nibble, but here shifted right, as by C-macro SPARNUM)
SPARITY_NONE    EQU     $00
SPARITY_EVEN    EQU     $01
SPARITY_ODD     EQU     $02

; Serial Handshake Mode (low nibble, mask by SHSHAKE_BITS)
SHSHAKE_XON     EQU     $00
SHSHAKE_RTS     EQU     $01
SHSHAKE_NONE    EQU     $02

; new defines for PrintFlags
CORRECT_RED         EQU $0001    ; color correct red shades
CORRECT_GREEN       EQU $0002    ; color correct green shades
CORRECT_BLUE        EQU $0004    ; color correct blue shades

CENTER_IMAGE        EQU $0008      ; center image on paper

IGNORE_DIMENSIONS   EQU $0000    ; ignore max width/height settings
BOUNDED_DIMENSIONS  EQU $0010    ; use max width/height as boundaries
ABSOLUTE_DIMENSIONS EQU $0020    ; use max width/height as absolutes
PIXEL_DIMENSIONS    EQU $0040    ; use max width/height as prt pixels
MULTIPLY_DIMENSIONS EQU $0080    ; use max width/height as multipliers

INTEGER_SCALING     EQU $0100    ; force integer scaling

ORDERED_DITHERING   EQU $0000    ; ordered dithering
HALFTONE_DITHERING  EQU $0200    ; halftone dithering
FLOYD_DITHERING     EQU $0400    ; floyd-steinberg dithering

ANTI_ALIAS          EQU $0800    ; anti-alias image
GREY_SCALE2         EQU $1000    ; for use with hi-res monitor

CORRECT_RGB_MASK    EQU (CORRECT_RED+CORRECT_GREEN+CORRECT_BLUE)
DIMENSIONS_MASK     EQU (BOUNDED_DIMENSIONS+ABSOLUTE_DIMENSIONS+PIXEL_DIMENSIONS
   +MULTIPLY_DIMENSIONS)
DITHERING_MASK      EQU (HALFTONE_DITHERING+FLOYD_DITHERING)

    ENDC
```

intuition/screens.i Page 1

```
        IFND  INTUITION_SCREENS_I
INTUITION_SCREENS_I     SET  1
**
**      $Filename: intuition/screens.i $
**      $Release: 2.04 $
**      $Revision: 36.17 $
**      $Date: 91/02/13 $
**
**      The Screen and NewScreen structures and attributes
**
**      (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**
        IFND EXEC_TYPES_I
        INCLUDE "exec/types.i"
        ENDC

        IFND GRAPHICS_GFX_I
        INCLUDE "graphics/gfx.i"
        ENDC

        IFND GRAPHICS_CLIP_I
        INCLUDE "graphics/clip.i"
        ENDC

        IFND GRAPHICS_VIEW_I
        INCLUDE "graphics/view.i"
        ENDC

        IFND GRAPHICS_RASTPORT_I
        INCLUDE "graphics/rastport.i"
        ENDC

        IFND GRAPHICS_LAYERS_I
        INCLUDE "graphics/layers.i"
        ENDC

        IFND UTILITY_TAGITEM_I
        INCLUDE "utility/tagitem.i"
        ENDC

*
* NOTE:  intuition/iobsolete.i is included at the END of this file!
*

; ======================================================================
; === DrawInfo =======================================================
; ======================================================================

* This is a packet of information for graphics rendering.  It originates
* with a Screen, and is gotten using GetScreenDrawInfo( screen );

* If you find dri_Version >= DRI_VERSION, you know this structure
* has at least the fields defined in this version of the include file

DRI_VERSION     EQU     1

 STRUCTURE DrawInfo,0
    UWORD       dri_Version       ; will be  DRI_VERSION
    UWORD       dri_NumPens       ; guaranteed to be >= NUMDRIPENS
    APTR        dri_Pens          ; pointer to pen array
    APTR        dri_Font          ; screen default font
    UWORD       dri_Depth         ; (initial) depth of screen bitmap
    ; from DisplayInfo database for initial display mode
    UWORD       dri_ResolutionX
    UWORD       dri_ResolutionY
    ULONG       dri_Flags
   STRUCT       dri_longreserved,28

DRIF_NEWLOOK    EQU     $00000001 ; specified SA_Pens, full treatment
DRIB_NEWLOOK    EQU     0

    ; rendering pen number indexes into DrawInfo.dri_Pens[]
    ENUM
    EITEM       DETAILPEN      ; compatible Intuition rendering pens
    EITEM       BLOCKPEN,
    EITEM       TEXTPEN        ; text on background (pen = 0)
    EITEM       SHINEPEN       ; bright edge on bas-relief
    EITEM       SHADOWPEN      ; dark edge
    EITEM       FILLPEN        ; active window fill
    EITEM       FILLTEXTPEN    ; text over FILLPEN
    EITEM       BACKGROUNDPEN  ; always color 0
    EITEM       HIGHLIGHTTEXTPEN  ; highlighted text, against BACKGROUNDPEN
    EITEM       NUMDRIPENS


; ======================================================================
; === Screen ============================================================
; ======================================================================
 STRUCTURE Screen,0

    APTR sc_NextScreen         ; linked list of screens
    APTR sc_FirstWindow        ; linked list Screen's Windows

    WORD sc_LeftEdge           ; parameters of the screen
    WORD sc_TopEdge            ; parameters of the screen

    WORD sc_Width              ; null-terminated Title text
    WORD sc_Height             ; for Windows without ScreenTitle

    WORD sc_MouseY             ; position relative to upper-left
    WORD sc_MouseX             ; position relative to upper-left

    WORD sc_Flags              ; see definitions below

    APTR sc_Title
    APTR sc_DefaultTitle

    ; Bar sizes for this Screen and all Window's in this Screen
    BYTE sc_BarHeight
    BYTE sc_BarVBorder
    BYTE sc_BarHBorder
    BYTE sc_MenuVBorder
    BYTE sc_MenuHBorder
    BYTE sc_WBorTop
    BYTE sc_WBorLeft
    BYTE sc_WBorRight
    BYTE sc_WBorBottom

    BYTE sc_KludgeFill00       ; This is strictly for word-alignment

    ; the display data structures for this Screen
    APTR sc_Font                       ; this screen's default font
    STRUCT sc_ViewPort,vp_SIZEOF       ; describing the Screen's display
    STRUCT sc_RastPort,rp_SIZEOF       ; describing Screen rendering
    STRUCT sc_BitMap,bm_SIZEOF         ; auxiliary graphexcess baggage
    STRUCT sc_LayerInfo,li_SIZEOF      ; each screen gets a LayerInfo

    APTR sc_FirstGadget

    BYTE sc_DetailPen          ; for bar/border/gadget rendering
    BYTE sc_BlockPen           ; for bar/border/gadget rendering
```

```

    ; the following variable(s) are maintained by Intuition to support the
    ; DisplayBeep() color flashing technique
    WORD sc_SaveColor0

    ; This layer is for the Screen and Menu bars
    APTR sc_BarLayer           ; was "BarLayer"

    APTR sc_ExtData

    APTR sc_UserData           ; general-purpose pointer to User data

    LABEL sc_SIZEOF     ; actually, you have no business talking about
                        ; or relying on the size of a screen structure

; --- FLAGS SET BY INTUITION ------------------------------------------------
; The SCREENTYPE bits are reserved for describing various Screen types
; available under Intuition.
SCREENTYPE      EQU     $000F   ; all the screens types available
; --- the definitions for the Screen Type -----------------------------------
WBENCHSCREEN    EQU     $0001   ; identifies the Workbench screen
PUBLICSCREEN    EQU     $0002   ; public shared (custom) screen
CUSTOMSCREEN    EQU     $000F   ; for that special look

SHOWTITLE       EQU     $0010   ; this gets set by a call to ShowTitle()

BEEPING EQU     $0020   ; set when Screen is beeping

CUSTOMBITMAP    EQU     $0040   ; if you are supplying your own BitMap

SCREENBEHIND    EQU     $0080   ; if you want your screen to open behind
                                ; already open screens

SCREENQUIET     EQU     $0100   ; if you do not want Intuition to render
                                ; into your screen (gadgets, title)

SCREENHIRES     EQU     $0200   ; do not use lowres gadgets (set by intuition)

STDSCREENHEIGHT EQU     -1      ; supply in NewScreen.Height
STDSCREENWIDTH  EQU     -1      ; supply in NewScreen.Width

NS_EXTENDED     EQU     $1000   ; means ns_Extenion is valid
AUTOSCROLL      EQU     $4000   ; automatic scrolling of large raster

* Screen attribute tag ID's.  These are used in the ti_Tag field of
* TagItem arrays passed to OpenScreenTagList() (or in the
* ExtNewScreen.Extension field).

* Screen attribute tags.  Please use these versions, not those in
* iobsolete.h.

 ENUM TAG_USER+33
*
*   these items specify items equivalent to fields in NewScreen
    EITEM SA_Left       ; traditional screen positions  and dimensions
    EITEM SA_Top
    EITEM SA_Width
    EITEM SA_Height
    EITEM SA_Depth      ; screen bitmap depth
    EITEM SA_DetailPen  ; serves as default for windows, too
    EITEM SA_BlockPen
    EITEM SA_Title      ; default screen title

    EITEM SA_Colors     ; ti_Data is an array of struct ColorSpec,
                        ; terminated by ColorIndex = -1.  Specifies
                        ; initial screen palette colors.

    EITEM SA_ErrorCode  ; ti_Data points to LONG error code (values below)
    EITEM SA_Font       ; equiv. to NewScreen.Font
    EITEM SA_SysFont    ; Selects one of the preferences system fonts:
                        ;       0 - old DefaultFont, fixed-width
                        ;       1 - WB Screen preferred font


    EITEM SA_Type       ; equiv. to NewScreen.Type
    EITEM SA_BitMap     ; ti_Data is pointer to custom BitMap.  This
                        ; implies type of CUSTOMBITMAP

    EITEM SA_PubName    ; presence of this tag means that the screen
                        ; is to be a public screen.  Please specify
                        ; BEFORE the two tags below

    EITEM SA_PubSig
    EITEM SA_PubTask    ; Task ID and signal for being notified that
                        ; the last window has closed on a public screen.


    EITEM SA_DisplayID  ; ti_Data is new extended display ID from
                        ; <graphics/displayinfo.h>.

    EITEM SA_DClip      ; ti_Data points to a rectangle which defines
                        ; screen display clip region

    EITEM SA_Overscan   ; was S_STDDCLIP.  Set to one of the OSCAN_
                        ; specifiers below to get a system standard
                        ; overscan region for your display clip,
                        ; screen dimensions (unless otherwise specified),
                        ; and automatically centered position (partial
                        ; support only so far).

    EITEM SA_Obsolete1  ; obsolete S_MONITORNAME

*   booleans *
    EITEM SA_ShowTitle  ; boolean equivalent to flag SHOWTITLE
    EITEM SA_Behind     ; boolean equivalent to flag SCREENBEHIND
    EITEM SA_Quiet      ; boolean equivalent to flag SCREENQUIET
    EITEM SA_AutoScroll ; boolean equivalent to flag AUTOSCROLL
    EITEM SA_Pens       ; array as in DrawInfo, terminated by -1
    EITEM SA_FullPalette ; boolean: initialize color table to entire
                         ;  preferences palette (32 for V36), rather
                         ; than compatible pens 0-3, 17-19, with
                         ; remaining palette as returned by GetColorMap()


* OpenScreen error codes, which are returned in the (optional) LONG
* pointed to by ti_Data for the SA_ErrorCode tag item

OSERR_NOMONITOR EQU     (1)     ; named monitor spec not available
OSERR_NOCHIPS   EQU     (2)     ; you need newer custom chips
OSERR_NOMEM     EQU     (3)     ; couldn't get normal memory
OSERR_NOCHIPMEM EQU     (4)     ; couldn't get chipmem
OSERR_PUBNOTUNIQUE      EQU (5) ; public screen name already used
OSERR_UNKNOWNMODE       EQU (6) ; don't recognize mode asked for

; =======================================================================
; === NewScreen =========================================================
; =======================================================================
; NOTE: to use Extension field, you need to use ExtNewScreen, below
 STRUCTURE NewScreen,0

    WORD ns_LeftEdge            ; initial Screen dimensions
```

```
    WORD ns_TopEdge             ; initial Screen dimensions
    WORD ns_Width               ; initial Screen dimensions
    WORD ns_Height              ; initial Screen dimensions
    WORD ns_Depth               ; initial Screen dimensions

    BYTE ns_DetailPen           ; default rendering pens (for Windows too)
    BYTE ns_BlockPen            ; default rendering pens (for Windows too)

    WORD ns_ViewModes           ; display "modes" for this Screen

    WORD ns_Type                ; Intuition Screen Type specifier

    APTR ns_Font                ; default font for Screen and Windows

    APTR ns_DefaultTitle        ; Title when Window doesn't care

    APTR ns_Gadgets             ; UNUSED:  Leave this NULL

    ; if you are opening a CUSTOMSCREEN and already have a BitMap
    ; that you want used for your Screen, you set the flags CUSTOMBITMAP in
    ; the Types variable and you set this variable to point to your BitMap
    ; structure.  The structure will be copied into your Screen structure,
    ; after which you may discard your own BitMap if you want
    APTR ns_CustomBitMap
 LABEL    ns_SIZEOF

; For compatibility reasons, we need a new structure for extending
; NewScreen.  Use this structure is you need to use the new Extension
; field.
; NOTE WELL: this structure may be extended again in the future.
;Writing code which depends on its size is not allowed.

 STRUCTURE ExtNewScreen,ns_SIZEOF

    APTR ens_Extension          ; struct TagItem *
                                ; more specification data, scanned if
                                ; NS_EXTENDED is set in ns_Type

 LABEL    ens_SIZEOF

* === Overscan Types ===
OSCAN_TEXT      EQU     1       ; entirely visible
OSCAN_STANDARD  EQU     2       ; just past edges
OSCAN_MAX       EQU     3       ; as much as possible
OSCAN_VIDEO     EQU     4       ; even more than is possible


* === Public Shared Screen Node ===

* This is the representative of a public shared screen.
* This is an internal data structure, but some functions may
* present a copy of it to the calling application.  In that case,
* be aware that the screen pointer of the structure can NOT be
* used safely, since there is no guarantee that the referenced
* screen will remain open and a valid data structure.

 STRUCTURE PubScreenNode,LN_SIZE
    APTR        psn_Screen      ; pointer to screen itself
    UWORD       psn_Flags       ; below
    WORD        psn_Size        ; includes name buffer size
    WORD        psn_VisitorCount ; how many visitor windows
    APTR        psn_SigTask     ; who to signal when visitors gone
    UBYTE       psn_SigBit      ; which signal
    UBYTE       psn_Pad1        ; word align
 LABEL          psn_SIZEOF
```

intuition/screens.i Page 6

```
* psn_Flags values
PSNF_PRIVATE    EQU     $0001

MAXPUBSCREENNAME EQU    139     ; names no longer, please

; pub screen modes
SHANGHAI        EQU     $0001   ; put workbench windows on pub screen
POPPUBSCREEN    EQU     $0002   ; pop pub screen to front when visitor opens

* Include obsolete identifiers:
        IFND    INTUITION_IOBSOLETE_I
        INCLUDE "intuition/iobsolete.i"
        ENDC

        ENDC
```

intuition/sghooks.i Page 1

```
    IFND INTUITION_SGHOOKS_I
INTUITION_SGHOOKS_I SET 1
**
** $Filename: intuition/sghooks.i $
** $Release: 2.04 $
** $Revision: 36.4 $
** $Date: 91/02/12 $
**
**  String gadget extensions and hooks
**
**  (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**

        IFND EXEC_TYPES_I
        INCLUDE "exec/types.i"
        ENDC

 STRUCTURE StringExtend,0
    ; display specifications
    APTR        sex_Font        ; must be an open Font (not TextAttr)
    STRUCT      sex_Pens,2      ; color of text/background
    STRUCT      sex_ActivePens,2 ; colors when gadget is active

    ; edit specifications
    ULONG       sex_InitialModes ; initial mode flags, below
    APTR        sex_EditHook    ; if non-NULL, must supply WorkBuffer
    APTR        sex_WorkBuffer  ; must be as large as StringInfo.Buffer

    STRUCT      sex_Reserved,16 ; set to 0
 LABEL  sex_SIZEOF


 STRUCTURE SGWork,0
    ; set up when gadget is first activated
    APTR                sgw_Gadget      ; the contestant itself
    APTR                sgw_StringInfo  ; easy access to sinfo
    APTR                sgw_WorkBuffer  ; intuition's planned result
    APTR                sgw_PrevBuffer  ; what was there before
    ULONG               sgw_Modes       ; current mode

    ; modified for each input event
    APTR                sgw_IEvent      ; actual event: do not change
    UWORD               sgw_Code        ; character code, if one byte
    WORD                sgw_BufferPos   ; cursor position
    WORD                sgw_NumChars
    ULONG               sgw_Actions     ; what Intuition will do
    LONG                sgw_LongInt     ; temp storage for longint

    APTR                sgw_GadgetInfo  ; see cghooks.h
    UWORD               sgw_EditOp      ; from constants below
 LABEL  sgw_SIZEOF      ; this may change as the structure is extended


* SGWork.EditOp -
* These values indicate what basic type of operation the global
* editing hook has performed on the string before your gadget's custom
* editing hook gets called.  You do not have to be concerned with the
* value your custom hook leaves in the sgw_EditOp field, only if you
* write a global editing hook.
*
* For most of these general edit operations, you'll want to compare
* the BufferPos and NumChars of the StringInfo (before global editing)
* and SGWork (after global editing).

EO_NOOP         EQU ($0001)
```

```
        ; did nothing
EO_DELBACKWARD  EQU ($0002)
        ; deleted some chars (maybe 0).
EO_DELFORWARD   EQU ($0003)
        ; deleted some characters under and in front of the cursor
EO_MOVECURSOR   EQU ($0004)
        ; moved the cursor
EO_ENTER        EQU ($0005)
        ; "enter" or "return" key, terminate
EO_RESET        EQU ($0006)
        ; current Intuition-style undo
EO_REPLACECHAR  EQU ($0007)
        ; replaced one character and (maybe) advanced cursor
EO_INSERTCHAR   EQU ($0008)
        ; inserted one char into string or added one at end
EO_BADFORMAT    EQU ($0009)
        ; didn't like the text data, e.g., Bad LONGINT
EO_BIGCHANGE    EQU ($000A)     ; unused by Intuition
        ; complete or major change to the text, e.g. new string
EO_UNDO         EQU ($000B)     ; unused by Intuition
        ; some other style of undo
EO_CLEAR        EQU ($000C)
        ; clear the string
EO_SPECIAL      EQU ($000D)     ; unused by Intuition
        ; some operation that doesn't fit into the categories here


; Mode Flags definitions (ONLY first group allowed as InitialModes)

SGM_REPLACE     EQU     1
SGMB_REPLACE    EQU     0
SGMF_REPLACE    EQU     1
; please initialize StringInfo with in-range value of BufferPos
; if you are using SGM_REPLACE mode.

; fixed length buffer, always set SGM_REPLACE, too.
SGM_FIXEDFIELD  EQU $00000002
SGMB_FIXEDFIELD EQU 1
SGMF_FIXEDFIELD EQU 2

; don't filter control chars
SGM_NOFILTER    EQU $00000004
SGMB_NOFILTER   EQU 2
SGMF_NOFILTER   EQU 4

; SGM_EXITHELP is new for V37, and ignored by V36:
; exit with code = 0x5F if HELP hit
SGM_EXITHELP    EQU $00000080
SGMB_EXITHELP   EQU 7
SGMF_EXITHELP   EQU $80


; String Gadget Action Flags (put in SGWork.Actions by EditHook)
SGA_USE         EQU $1          ; use contents of SGWork
SGAB_USE        EQU 0
SGAF_USE        EQU $1

SGA_END         EQU $2          ; terminate gadget, code in Code field
SGAB_END        EQU 1
SGAF_END        EQU $2

SGA_BEEP        EQU $4          ; flash the screen for the user
SGAB_BEEP       EQU 2
SGAF_BEEP       EQU $4

SGA_REUSE       EQU $8          ; reuse input event
```

```
SGAB_REUSE      EQU 3
SGAF_REUSE      EQU $8

SGA_REDISPLAY   EQU $10         ; gadget visuals changed
SGAB_REDISPLAY          EQU 4
SGAF_REDISPLAY          EQU $10

; New for V37:
SGA_NEXTACTIVE  EQU $20         ; Make next possible gadget active.
SGAB_NEXTACTIVE EQU 5
SGAF_NEXTACTIVE EQU $20

; New for V37:
SGA_PREVACTIVE  EQU $40         ; Make previous possible gadget active.
SGAB_PREVACTIVE EQU 6
SGAF_PREVACTIVE EQU $40


; function id for only existing custom string gadget edit hook
SGH_KEY         EQU     1       ; process editing keystroke
SGH_CLICK       EQU     2       ; process mouse click cursor position

* Here's a brief summary of how the custom string gadget edit hook works:
*       You provide a hook in StringInfo.Extension.EditHook.
*       The hook is called in the standard way with the 'object'
*       a pointer to SGWork, and the 'message' a pointer to a command
*       block, starting either with (longword) SGH_KEY, SGH_CLICK,
*       or something new.
*
*       You return 0 if you don't understand the command (SGH_KEY is
*       required and assumed).  Return non-zero if you implement the
*       command.
*
*   SGH_KEY:
*       There are no parameters following the command longword.
*
*       Intuition will put its idea of proper values in the SGWork
*       before calling you, and if you leave SGA_USE set in the
*       SGWork.Actions field, Intuition will use the values
*       found in SGWork fields WorkBuffer, NumChars, BufferPos,
*       and LongInt, copying the WorkBuffer back to the StringInfo
*       Buffer.
*
*       NOTE WELL: You may NOT change other SGWork fields.
*
*       If you clear SGA_USE, the string gadget will be unchanged.
*
*       If you set SGA_END, Intuition will terminate the activation
*       of the string gadget.  If you also set SGA_REUSE, Intuition
*       will reuse the input event after it deactivates your gadget.
*
*       In this case, Intuition will put the value found in SGWork.Code
*       into the IntuiMessage.Code field of the IDCMP_GADGETUP message it
*       sends to the application.
*
*       If you set SGA_BEEP, Intuition will call DisplayBeep(); use
*       this if the user has typed in error, or buffer is full.
*
*       Set SGA_REDISPLAY if the changes to the gadget warrant a
*       gadget redisplay.  Note: cursor movement requires a redisplay.
*
*       Starting in V37, you may set SGA_PREVACTIVE or SGA_NEXTACTIVE
*       when you set SGA_END.  This tells Intuition that you want
*       the next or previous gadget with GFLG_TABCYCLE to be activated.
*
*   SGH_CLICK:
```

```
*       This hook command is called when Intuition wants to position
*       the cursor in response to a mouse click in the string gadget.
*
*       Again, here are no parameters following the command longword.
*
*       This time, Intuition has already calculated the mouse position
*       character cell and put it in SGWork.BufferPos.  The previous
*       BufferPos value remains in the SGWork.StringInfo.BufferPos.
*
*       Intuition will again use the SGWork fields listed above for
*       SGH_KEY.  One restriction is that you are NOT allowed to set
*       SGA_END or SGA_REUSE for this command.  Intuition will not
*       stand for a gadget which goes inactive when you click in it.
*
*       You should always leave the SGA_REDISPLAY flag set, since Intuition
*       uses this processing when activating a string gadget.

    ENDC
```

libraries/asl.i Page 1

```
        IFND LIBRARIES_ASL_I
LIBRARIES_ASL_I SET     1
**
**      $Filename: libraries/asl.i $
**      $Release: 2.04 $
**      $Revision: 36.4 $
**      $Date: 91/03/06 $
**
**      ASL library name and useful definitions.
**
**      (C) Copyright 1989,1990 Commodore-Amiga Inc. and Charlie Heath
**              All Rights Reserved
**

        IFND EXEC_TYPES_I
        INCLUDE "exec/types.i"
        ENDC

        IFND EXEC_LIBRARIES_I
        INCLUDE "exec/libraries.i"
        ENDC

        IFND EXEC_LISTS_I
        INCLUDE "exec/lists.i"
        ENDC

        IFND    UTILITY_TAGITEM_I
        INCLUDE "utility/tagitem.i"
        ENDC

*************************************************************************

AslName MACRO
        dc.b    'asl.library',0
        ds.w    0
        ENDM


*************************************************************************
* REQUESTER TYPES, these are passed to AllocAslRequest
*************************************************************************

ASL_FileRequest         equ     0
ASL_FontRequest         equ     1


**************************************************************************
*                                                                        *
*       The ASL file requester data structure...                         *
*                                                                        *
* The fields described here are for READ ACCESS to the structure         *
* returned by AllocAslRequest( ASL_FileRequest, ... )                    *
*                                                                        *
* Any modifications MUST be done via TAGS either at the time of          *
* creation by AllocAslRequest(), or when used, via AslRequest()          *
*                                                                        *
**************************************************************************

        STRUCTURE FileRequester,0
        STRUCTURE FileRequestr,0                ; obsolete spelling
                CPTR    rf_Reserved1
                CPTR    rf_File         ; *Filename array (FCHARS+1)
                CPTR    rf_Dir                  ; *Directory array (DSIZE+1)
                CPTR    rf_Reserved2
                UBYTE   rf_Reserved3
                UBYTE   rf_Reserved4
```

libraries/asl.i Page 2

```
                APTR    rf_Reserved5
                WORD    rf_LeftEdge
                WORD    rf_TopEdge
                WORD    rf_Width
                WORD    rf_Height
                WORD    rf_Reserved6
                LONG    rf_NumArgs
                APTR    rf_ArgList
                APTR    rf_UserData
                APTR    rf_Reserved7
                APTR    rf_Reserved8
                CPTR    rf_Pat                  ; *Pattern array

******************************************************************************
*
* The following defined values are the ASL_FuncFlags tag values which
* are defined for the ASL file request.  These values may be passed
* as a TagItem to modify the way the requester is presented.  Each
* flag value defined has a description of the particular action.
*
* Also related to the ASL_FuncFlags values is the ASL_HookFunc tagitem,
* which provides a callback function to allow the application to
* interact with the requester.  If an ASL_HookFunc TagItem is
* defined, that function will be called as follows:
*
* ULONG rf_Function(ULONG Mask, CPTR Object, CPTR AslRequester)
*
* The Mask value is a copy of the specific ASL_FuncFlags value
* the callback is for; Object is a pointer to a data object.
* AslRequester is a pointer to the requester structure.
*
* For the ASL file and font requesters, two ASL_FuncFlags values
* are currently defined; FILF_DOWILDFUNC and FILF_DOMSGFUNC.
*
******************************************************************************

* Pass these flags with the tag ASL_FuncFlags
        BITDEF  FIL,PATGAD,0            ; Request a pattern gadget
        BITDEF  FIL,MULTISELECT,3       ; Request multiple selection returns -
                                        ; MUTUAL EXCLUSIVE WITH SAVE
        BITDEF  FIL,NEWIDCMP,4  ; Force a new IDCMP (only if rf_Window != NULL)
        BITDEF  FIL,SAVE,5      ; Use this bit for SAVE requesters
        BITDEF  FIL,DOMSGFUNC,6 ; Called with Object=IDCMP messages
                                ;  for other windows of shered port.
                                ;  You must return pointer to Object,
                                ;  asl will reply the Object for you.
        BITDEF  FIL,DOWILDFUNC,7 ; Called with an AnchorPath,
                                 ;      ZERO return accepts.


* Pass these flags with the tag ASL_ExtFlags
        BITDEF  FIL1,NOFILES,0    ; Do not want a file gadget, no files shown
        BITDEF  FIL1,MATCHDIRS,1  ; Patgad/rf_Pat should screen files AND DIRS

******************************************************************************
* Obsolete - Use FIL flag names instead
        BITDEF  RF,DOWILDFUNC,7 ; Called me with an AnchorPath,
                                ;       ZERO return accepts.
        BITDEF  RF,DOMSGFUNC,6  ; You get all IDCMP message not for FileRequest(
)
        BITDEF  RF,DOCOLOR,5    ; This bit is used for FILE SAVE operations.
        BITDEF  RF,NEWIDCMP,4   ; Force a new IDCMP (only if rf_Window != NULL)
        BITDEF  RF,MULTISELECT,3 ; Request multiple selection returns -
                                ;       MUTUAL EXCLUSIVE WITH DOCOLOR
        BITDEF  RF,PATGAD,0     ; Request a pattern gadget
******************************************************************************
```

```
    *

        STRUCTURE FontRequester,0
                CPTR    fo_Reserved1
                CPTR    fo_Reserved2
                APTR    fo_Name         ; Returned name
                USHORT  fo_YSize
                UBYTE   fo_Style
                UBYTE   fo_Flags
                UBYTE   fo_FrontPen
                UBYTE   fo_BackPen
                UBYTE   fo_DrawMode
                UBYTE   fo_Reserved3

                APTR    fo_UserData

                SHORT   fo_LeftEdge
                SHORT   fo_TopEdge

                SHORT   fo_Width
                SHORT   fo_Height


******* BITDEFS for ASL_FuncFlags - FONT requester

        BITDEF  FON,FRONTCOLOR,0        ; Display Front Color palette selector?
        BITDEF  FON,BACKCOLOR,1         ; Display Back Color palette selector?
        BITDEF  FON,STYLES,2            ; Display Styles checkboxes?
        BITDEF  FON,DRAWMODE,3          ; Display DrawMode NWAY selector?
        BITDEF  FON,FIXEDWIDTH,4        ; Only allow fixed-width (SYS) fonts?
        BITDEF  FON,NEWIDCMP,5          ; Request a NEW IDCMP port,
                                        ;       rather than shared.
        BITDEF  FON,DOMSGFUNC,6         ; Called with Object=IDCMP message
                                        ;  for other windows sharing port.
                                        ; You must return pointer to Object.
                                        ; asl will reply the object for you.
        BITDEF  FON,DOWILDFUNC,7        ; Called with Object=TextAttr
                                        ;       NON-Zero return accepts

****    Tag values for AslRequest()     *************************************

ASL_Dummy       equ     TAG_USER+$80000

ASL_Hail        equ     ASL_Dummy+1
ASL_Window      equ     ASL_Dummy+2

ASL_LeftEdge    equ     ASL_Dummy+3
ASL_TopEdge     equ     ASL_Dummy+4
ASL_Width       equ     ASL_Dummy+5
ASL_Height      equ     ASL_Dummy+6

ASL_HookFunc    equ     ASL_Dummy+7

ASL_File        equ     ASL_Dummy+8
ASL_Dir         equ     ASL_Dummy+9

* OVERLAP HERE file and font stuffskies!!!
ASL_Pattern     equ     ASL_Dummy+10

* Font specific, some overlap!!!
ASL_FontName    equ     ASL_Dummy+10
ASL_FontHeight  equ     ASL_Dummy+11
ASL_FontStyles  equ     ASL_Dummy+12
ASL_FontFlags   equ     ASL_Dummy+13
ASL_FrontPen    equ     ASL_Dummy+14
ASL_BackPen     equ     ASL_Dummy+15
```

```
ASL_MinHeight   equ     ASL_Dummy+16
ASL_MaxHeight   equ     ASL_Dummy+17

ASL_OKText      equ     ASL_Dummy+18
ASL_CancelText  equ     ASL_Dummy+19

ASL_FuncFlags   equ     ASL_Dummy+20

ASL_ModeList    equ     ASL_Dummy+21

*** Pass the FIL1 extended flag bits using this tag
ASL_ExtFlags1   equ     ASL_Dummy+22

        ENDC    !LIBRARIES_ASL_I
```

libraries/commodities.i Page 1

```
        IFND LIBRARIES_COMMODITIES_I
LIBRARIES_COMMODITIES_I SET 1
**   $Filename: libraries/commodities.i $
**   $Release: 2.04 $
**   $Revision: 1.2 $
**   $Date: 91/02/22 $
**
**   Commodities definitions
**
**   (C) Copyright 1988,1989,1990 Commodore-Amiga Inc.
**      All Rights Reserved
*-----------------------------------------------------------*

        IFND EXEC_TYPES_I
        INCLUDE "exec/types.i"

        ENDC
        IFND DEVICES_INPUTEVENT_I
        INCLUDE "devices/inputevent.i"
        ENDC


***************
* Broker stuff
***************

* buffer sizes
CBD_NAMELEN     EQU     24
CBD_TITLELEN    EQU     40
CBD_DESCRLEN    EQU     40

* CxBroker errors
CBERR_OK        EQU     0       ; No error
CBERR_SYSERR    EQU     1       ; System error , no memory, etc
CBERR_DUP       EQU     2       ; uniqueness violation
CBERR_VERSION   EQU     3       ; didn't understand nb_VERSION
NB_VERSION      EQU     5       ; Version of NewBroker structure

        STRUCTURE NewBroker,0
           BYTE         nb_Version        ; set to NB_VERSION
           BYTE         nb_Reservel     ; for alignment
           APTR         nb_Name
           APTR         nb_Title
           APTR         nb_Descr
           WORD         nb_Unique
           WORD         nb_Flags
           BYTE         nb_Pri            ; new in V5
           BYTE         nb_Reserve2     ; for alignment
           APTR         nb_Port
           WORD         nb_ReservedChannel      ;plans for later port sharing

***********************
* Flags for nb_Unique
***********************

NBU_DUPLICATE   EQU       0
NBU_UNIQUE      EQU       1       ; will not allow duplicates
NBU_NOTIFY      EQU       2       ; sends CXM_UNIQUE to existing broker

* Flags for nb_Flags
COF_SHOW_HIDE   EQU       4

********
* cxusr
********

```

libraries/commodities.i Page 2

```
*********************************
** Commodities Object Types   **
*********************************
CX_INVALID      EQU     0     ; not a valid object (probably null)
CX_FILTER       EQU     1     ; input event messages only
CX_TYPEFILTER   EQU     2     ; filter on message type
CX_SEND         EQU     3     ; sends a message
CX_SIGNAL       EQU     4     ; sends a signal
CX_TRANSLATE    EQU     5     ; translates IE into chain
CX_BROKER       EQU     6     ; application representative
CX_DEBUG        EQU     7     ; dumps kprintf to serial port
CX_CUSTOM       EQU     8     ; application provids function
CX_ZERO         EQU     9     ; system terminator node

*****************
** CxMsg types **
*****************
CXM_UNIQUE      EQU        $10 ; (1 << 4) sent down broker by CxBroker()
; Obsolete: subsumed by CXM_COMMAND (below)

* Messages of this type rattle around the Commodities input network.
* They will be sent to you by a Sender object, and passed to you
* as a synchronous function call by a Custom object.
*
* The message port or function entry point is stored in the object,
* and the ID field of the message will be set to what you arrange
* issuing object.
*
* The Data field will point to the input event triggering the
* message.

CXM_IEVENT      EQU        $20 ;(1 << 5)

* These messages are sent to a port attached to your Broker.
* They are sent to you when the controller program wants your
* program to do something.  The ID field identifies the command.
*
* The Data field will be used later.
*

CXM_COMMAND     EQU        $40 ;(1 << 6)

* ID values

CXCMD_DISABLE   EQU     15  ; please disable yourself
CXCMD_ENABLE    EQU     17  ; please enable yourself
CXCMD_APPEAR    EQU     19  ; open your window, if you can
CXCMD_DISAPPEAR EQU     21  ; go dormant
CXCMD_KILL      EQU     23  ; go away for good
CXCMD_UNIQUE    EQU     25  ; someone tried to create a brok
                            ; with your name.  Suggest you Appear.
CXCMD_LIST_CHG  EQU     27  ; Used by Exchange program. Someone
                            ; has changed the broker list

* return values for BrokerCommand()
CMDE_OK         EQU       0
CMDE_NOBROKER   EQU      -1
CMDE_NOPORT     EQU      -2
CMDE_NOMEM      EQU      -3

* IMPORTANT NOTE: for V5:
* Only CXM_IEVENT messages are passed through the input network.
*
* Other types of messages are sent to an optional port in your broker.
*
* This means that you must test the message type in your message handling,
```

libraries/commodities.i

```
* if input messages and command messages come to the same port.
*
* Older programs have no broker port, so processing loops which
* make assumptions about type won't encounter the new message types.
*
* The TypeFilter CxObject is hereby obsolete.
*
* It is less convenient for the application, but eliminates testing
* for type of input messages.

************************************************************
** CxObj Error Flags (return values from CxObjError())  **
************************************************************

COERR_ISNULL            EQU     1  ; you called CxError(NULL)
COERR_NULLATTACH        EQU     2  ; someone attached NULL to my list
COERR_BADFILTER EQU     4  ; a bad filter description was given
COERR_BADTYPE           EQU     8  ; unmatched type-specific operation


******************************
* Input Expression structure *
******************************

IX_VERSION      EQU       2

        STRUCTURE InputXpression,0

           UBYTE   ix_Version      ; must be set to IX_VERSION
           UBYTE   ix_Class        ; class must match exactly

           UWORD   ix_Code         ; Bits that we want

           UWORD   ix_CodeMask     ; Set bits here to indicate
                                   ; which bits in ix_Code are
                                   ; don't care bits.

           UWORD   ix_Qualifier    ; Bits that we want

           UWORD   ix_QualMask     ; Set bits here to indicate
                                   ; which bits in ix_Qualifier
                                   ; are don't care bits

           UWORD   ix_QualSame    ; synonyms in qualifier

        LABEL   ix_SIZEOF

* QualSame identifiers
IXSYM_SHIFT     EQU       1     ; left- and right- shift are equivalent
IXSYM_CAPS      EQU       2     ; either shift or caps lock are equivalent
IXSYM_ALT       EQU       4     ; left- and right- alt are equivalent

* corresponding QualSame masks
IXSYM_SHIFTMASK EQU      IEQUALIFIER_LSHIFT!IEQUALIFIER_RSHIFT
IXSYM_CAPSMASK          EQU      IXSYM_SHIFTMASK!IEQUALIFIER_CAPSLOCK
IXSYM_ALTMASK   EQU      IEQUALIFIER_LALT!IEQUALIFIER_RALT

IX_NORMALQUALS          EQU      $7FFF  ;for QualMask field: avoid RELATIVEMOUS
 E

        ENDC    ; LIBRARIES_COMMODITIES_I
```

libraries/configregs.i

```
        IFND    LIBRARIES_CONFIGREGS_I
LIBRARIES_CONFIGREGS_I  SET     1
**
**      $Filename: libraries/configregs.i $
**      $Release: 2.04 $
**      $Revision: 36.11 $
**      $Date: 90/11/03 $
**
**      AutoConfig (tm) hardware register and bit definitions
**
**      (C) Copyright 1985,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**


        IFND    EXEC_TYPES_I
        INCLUDE "exec/types.i"
        ENDC    ;EXEC_TYPES_I

**
** AutoConfig (tm) boards each contain a 32 byte "ExpansionRom" area that is
** read by the system software at configuration time.  Configuration of each
** board starts when the ConfigIn* signal is passed from the previous board
** (or from the system for the first board).  Each board will present it's
** ExpansionRom structure at location $00E80000 to be read by the system.
** This file defines the appearance of the ExpansionRom area.
**
** Expansion boards are actually organized such that only one nybble per
** 16 bit word contains valid information.  The low nybbles of each
** word are combined to fill the structure below. (This table is structured
** as LOGICAL information.  This means that it never corresponds exactly
** with a physical implementation.)
**
** The ExpansionRom space is further split into two regions:  The first 16
** bytes are read-only.  Except for the er_type field, this area is inverted
** by the system software when read in.  The second 16 bytes contain the
** control portion, where all read/write registers are located.
**
** The system builds one "ConfigDev" structure for each board found.  The
** list of boards can be examined using the expansion.library/FindConfigDev
** function.
**
** A special "hacker" Manufacturer ID number is reserved for test use:
** 2011 ($7DB).  When inverted this will look like $F824.
**

 STRUCTURE ExpansionRom,0        ;-First 16 bytes of the expansion ROM
    UBYTE       er_Type          ;Board type, size and flags
    UBYTE       er_Product       ;Product number, assigned by manufacturer
    UBYTE       er_Flags         ;Flags
    UBYTE       er_Reserved03    ;Must be zero ($ff inverted)
    UWORD       er_Manufacturer ;Unique ID,ASSIGNED BY COMMODORE-AMIGA!
    ULONG       er_SerialNumber ;Available for use by manufacturer
    UWORD       er_InitDiagVec  ;Offset to optional "DiagArea" structure
    UBYTE       er_Reserved0c
    UBYTE       er_Reserved0d
    UBYTE       er_Reserved0e
    UBYTE       er_Reserved0f
    LABEL       ExpansionRom_SIZEOF


**
** Note that use of the ec_BaseAddress register is tricky.  The system
** will actually write twice.  First the low order nybble is written
** to the ec_BaseAddress register+2 (D15-D12).  Then the entire byte is
** written to ec_BaseAddress (D15-D8).  This allows writing of a byte-wide
** address to nybble size registers.
```

```
**

 STRUCTURE ExpansionControl,0   ;-Second 16 bytes of the expansion ROM
    UBYTE      ec_Interrupt     ;Optional interrupt control register
    UBYTE      ec_Z3_HighBase   ;Zorro III   : Bits 24-31 of config address
    UBYTE      ec_BaseAddress   ;Zorro II/III: Bits 16-23 of config address
    UBYTE      ec_Shutup        ;The system writes here to shut up a board
    UBYTE      ec_Reserved14
    UBYTE      ec_Reserved15
    UBYTE      ec_Reserved16
    UBYTE      ec_Reserved17
    UBYTE      ec_Reserved18
    UBYTE      ec_Reserved19
    UBYTE      ec_Reserved1a
    UBYTE      ec_Reserved1b
    UBYTE      ec_Reserved1c
    UBYTE      ec_Reserved1d
    UBYTE      ec_Reserved1e
    UBYTE      ec_Reserved1f
    LABEL      ExpansionControl_SIZEOF

**
** many of the constants below consist of a triplet of equivalent
** definitions: xxMASK is a bit mask of those bits that matter.
** xxBIT is the starting bit number of the field.  xxSIZE is the
** number of bits that make up the definition.  This method is
** used when the field is larger than one bit.
**
** If the field is only one bit wide then the xxB_xx and xxF_xx convention
** is used (xxB_xx is the bit number, and xxF_xx is mask of the bit).
**

** manifest constants **
E_SLOTSIZE              EQU     $10000
E_SLOTMASK              EQU     $ffff
E_SLOTSHIFT             EQU     16


** these define the free regions of Zorro memory space.
** THESE MAY WELL CHANGE FOR FUTURE PRODUCTS!
E_EXPANSIONBASE         EQU     $00e80000       ;Zorro II  config address
EZ3_EXPANSIONBASE       EQU     $ff000000       ;Zorro III config address

E_EXPANSIONSIZE         EQU     $00080000       ;Zorro II  I/O type cards
E_EXPANSIONSLOTS        EQU     8

E_MEMORYBASE            EQU     $00200000       ;Zorro II  8MB space
E_MEMORYSIZE            EQU     $00800000
E_MEMORYSLOTS           EQU     128

EZ3_CONFIGAREA          EQU     $40000000       ;Zorro III space
EZ3_CONFIGAREAEND       EQU     $7FFFFFFF       ;Zorro III space
EZ3_SIZEGRANULARITY     EQU     $00080000       ;512K increments


***** er_Type definitions (ttldcmmm) ***************************************

** er_Type board type bits -- the OS ignores "old style" boards **
ERT_TYPEMASK            EQU     $c0     ;Bits 7-6
ERT_TYPEBIT             EQU     6
ERT_TYPESIZE            EQU     2
ERT_NEWBOARD            EQU     $c0
ERT_ZORROII             EQU     ERT_NEWBOARD
ERT_ZORROIII            EQU     $80

** other bits defined in er_Type **
```

```
        BITDEF  ERT,MEMLIST,5           ; Link RAM into free memory list
        BITDEF  ERT,DIAGVALID,4         ; ROM vector is valid
        BITDEF  ERT,CHAINEDCONFIG,3     ; Next config is part of the same card

** er_Type field memory size bits **
ERT_MEMMASK             EQU     $07     ;Bits 2-0
ERT_MEMBIT              EQU     0
ERT_MEMSIZE             EQU     3



***** er_Flags byte -- for those things that didn't fit into the type byte ****
***** the hardware stores this byte in inverted form                       ****
        BITDEF  ERF,MEMSPACE,7          ; Wants to be in 8 meg space.
                                        ; (NOT IMPLEMENTED)

        BITDEF  ERF,NOSHUTUP,6          ; Board can't be shut up.

        BITDEF  ERF,EXTENDED,5          ; Zorro III: Use extended size table
                                        ;            for bits 0-2 of er_Type.
                                        ; Zorro II : Must be 0

        BITDEF  ERF,ZORRO_III,4         ; Zorro III: must be 1
                                        ; Zorro II : must be 0

ERT_Z3_SSMASK           EQU     $0F     ; Bits 3-0.  Zorro III Sub-Size.  How
ERT_Z3_SSBIT            EQU     0       ; much space the card actually uses
ERT_Z3_SSSIZE           EQU     4       ; (regardless of config granularity)
                                        ; Zorro II : must be 0


** ec_Interrupt register (unused) ********************************************
        BITDEF  ECI,INTENA,1
        BITDEF  ECI,RESET,3
        BITDEF  ECI,INT2PEND,4
        BITDEF  ECI,INT6PEND,5
        BITDEF  ECI,INT7PEND,6
        BITDEF  ECI,INTERRUPTING,7


*****************************************************************************
**
** these are the specifications for the diagnostic area.  If the Diagnostic
** Address Valid bit is set in the Board Type byte (the first byte in
** expansion space) then the Diag Init vector contains a valid offset.
**
** The Diag Init vector is actually a word offset from the base of the
** board.  The resulting address points to the base of the DiagArea
** structure.  The structure may be physically implemented either four,
** eight, or sixteen bits wide.  The code will be copied out into
** ram first before being called.
**
** The da_Size field, and both code offsets (da_DiagPoint and da_BootPoint)
** are offsets from the diag area AFTER it has been copied into ram, and
** "de-nybbleized" (if needed). (In other words, the byte size is the size of
** the actual information, not how much address space is required to
** store it.)
**
** All bits are encoded with uninverted logic (e.g. 5 volts on the bus
** is a logic one).
**
** If your board is to make use of the boot facility then it must leave
** its config area available even after it has been configured.  Your
** boot vector will be called AFTER your board's final address has been
** set.
**
```

libraries/configregs.i

```
***************************************************************************

 STRUCTURE DiagArea,0
    UBYTE       da_Config       ; see below for definitions
    UBYTE       da_Flags        ; see below for definitions
    UWORD       da_Size ; the size (in bytes) of the total diag area
    UWORD       da_DiagPoint    ; where to start for diagnostics, or zero
    UWORD       da_BootPoint    ; where to start for booting
    UWORD       da_Name ; offset in diag area where a string
                                ;   identifier can be found (or zero if no
                                ;   identifier is present).

    UWORD       da_Reserved01   ; two words of reserved data.  must be zero.
    UWORD       da_Reserved02
    LABEL       DiagArea_SIZEOF

; da_Config definitions
**
** DAC_BYTEWIDE can be simulated using DAC_NIBBLEWIDE.
**
DAC_BUSWIDTH    EQU     $C0     ; two bits for bus width
DAC_NIBBLEWIDE  EQU     $00     ; (indicates information is nybble wide)
DAC_BYTEWIDE    EQU     $40     ; BUG: Will not work under V34 Kickstart!
DAC_WORDWIDE    EQU     $80

DAC_BOOTTIME    EQU     $30     ; two bits for when to boot
DAC_NEVER       EQU     $00     ; obvious
DAC_CONFIGTIME  EQU     $10     ; call da_BootPoint when first configing
                                ;   the device
DAC_BINDTIME    EQU     $20     ; run when binding drivers to boards

**
** These are the calling conventions for the diagnostic callback
** (from da_DiagPoint).
**
** A7 -- points to at least 2K of stack
** A6 -- ExecBase
** A5 -- ExpansionBase
** A3 -- your board's ConfigDev structure
** A2 -- Base of diag/init area that was copied
** A0 -- Base of your board
**
** Your board must return a value in D0.  If this value is NULL, then
** the diag/init area that was copied in will be returned to the free
** memory pool.
**

        ENDC    ;LIBRARIES_CONFIGREGS_I
```

---

libraries/configvars.i

```
        IFND    LIBRARIES_CONFIGVARS_I
LIBRARIES_CONFIGVARS_I  SET     1
**
**      $Filename: libraries/configvars.i $
**      $Release: 2.04 $
**      $Revision: 36.6 $
**      $Date: 90/05/08 $
**
**      Software structures used by AutoConfig (tm) boards
**
**      (C) Copyright 1985,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**

        IFND    EXEC_NODES_I
        INCLUDE "exec/nodes.i"
        ENDC    ;EXEC_NODES_I

        IFND    LIBRARIES_CONFIGREGS_I
        INCLUDE "libraries/configregs.i"
        ENDC    ;LIBRARIES_CONFIGREGS_I

**
** At early system startup time, one ConfigDev structure is created for
** each board found in the system.  Software may seach for ConfigDev
** structures by vendor & product ID number.  For debugging and diagnostic
** use, the entire list can be accessed.  See the expansion.library document
** for more information.
**
 STRUCTURE ConfigDev,0
    STRUCT      cd_Node,LN_SIZE
    UBYTE       cd_Flags        ; (read/write)
    UBYTE       cd_Pad          ; reserved
    STRUCT      cd_Rom,ExpansionRom_SIZEOF ; copy of board's expansion ROM
    APTR        cd_BoardAddr    ; where in memory the board was placed
    ULONG       cd_BoardSize    ; size of board in bytes
    UWORD       cd_SlotAddr     ; which slot number (PRIVATE)
    UWORD       cd_SlotSize     ; number of slots (PRIVATE)
    APTR        cd_Driver       ; pointer to node of driver
    APTR        cd_NextCD       ; linked list of drivers to config
    STRUCT      cd_Unused,4*4   ; for whatever the driver wants!
    LABEL       ConfigDev_SIZEOF

; cd_Flags
        BITDEF  CD,SHUTUP,0     ; this board has been shut up
        BITDEF  CD,CONFIGME,1   ; this board needs a driver to claim it
        BITDEF  CD,BADMEMORY,2  ; this board contains bad memory

**
** Boards are usually "bound" to software drivers.
** This structure is used by GetCurrentBinding() and SetCurrentBinding()
**
 STRUCTURE CurrentBinding,0
    APTR        cb_ConfigDev
    APTR        cb_FileName
    APTR        cb_ProductString
    APTR        cb_ToolTypes
    LABEL       CurrentBinding_SIZEOF

        ENDC    ;LIBRARIES_CONFIGVARS_I
```

libraries/diskfont.i

Page 1

```
        IFND    LIBRARIES_DISKFONT_I
LIBRARIES_DISKFONT_I    SET     1
**
**      $Filename: libraries/diskfont.i $
**      $Release: 2.04 $
**      $Revision: 36.3 $
**      $Date: 90/10/22 $
**
**      diskfont library definitions
**
**      (C) Copyright 1985,1986,1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

        IFND    EXEC_TYPES_I
        INCLUDE "exec/types.i"
        ENDC
        IFND    EXEC_NODES_I
        INCLUDE "exec/nodes.i"
        ENDC
        IFND    EXEC_LISTS_I
        INCLUDE "exec/lists.i"
        ENDC
        IFND    GRAPHICS_TEXT_I
        INCLUDE "graphics/text.i"
        ENDC

MAXFONTPATH     EQU     256     ; including null terminator

 STRUCTURE  FC,0                ; FontContents
    STRUCT  fc_FileName,MAXFONTPATH
    UWORD   fc_YSize
    UBYTE   fc_Style
    UBYTE   fc_Flags
    LABEL   fc_SIZEOF

 STRUCTURE  TFC,0               ; TFontContents
    STRUCT  tfc_FileName,MAXFONTPATH-2
    UWORD   tfc_TagCount        ; including the TAG_DONE tag
    ;
    ;   if tfc_TagCount is non-zero, tfc_FileName is overlayed with
    ;   Text Tags starting at:  (struct TagItem *)
    ;       &tfc_FileName[MAXFONTPATH-(tfc_TagCount*sizeof(struct TagItem))]
    ;
    UWORD   tfc_YSize
    UBYTE   tfc_Style
    UBYTE   tfc_Flags
    LABEL   tfc_SIZEOF


FCH_ID          EQU     $0f00   ; FontContentsHeader, then FontContents
TFCH_ID         EQU     $0f02   ; FontContentsHeader, then TFontContents


 STRUCTURE  FCH,0               ; FontContentsHeader
    UWORD   fch_FileID          ; FCH_ID or TFCH_ID
    UWORD   fch_NumEntries      ; the number of FontContents elements
    LABEL   fch_FC              ; the [T]FontContents elements follow here


DFH_ID          EQU     $0f80
MAXFONTNAME     EQU     32      ; font name including ".font\0"

 STRUCTURE  DiskFontHeader,0
    ; the following 8 bytes are not actually considered a part of the
    ; DiskFontHeader, but immediately preceed it.  The NextSegment is supplied
```

libraries/diskfont.i

Page 2

```
     ; by the linker/loader, and the ReturnCode is the code at the beginning
     ; of the font in case someone runs it...
     ; ULONG dfh_NextSegment     ; actually a BPTR
     ; ULONG dfh_ReturnCode      ; MOVEQ #0,D0 : RTS
     ; here then is the official start of the DiskFontHeader...
     STRUCT  dfh_DF,LN_SIZE      ; node to link disk fonts
     UWORD   dfh_FileID          ; DFH_ID
     UWORD   dfh_Revision        ; the font revision in this version
     LONG    dfh_Segment         ; the segment address when loaded
     STRUCT  dfh_Name,MAXFONTNAME ; the font name (null terminated)
     STRUCT  dfh_TF,tf_SIZEOF    ; loaded TextFont structure
     LABEL   dfh_SIZEOF

; used only if dfh_TF.tf_Style FSB_TAGGED bit is set
dfh_TagList     EQU     dfh_Segment     ; destroyed during loading


     BITDEF  AF,MEMORY,0
     BITDEF  AF,DISK,1
     BITDEF  AF,SCALED,2

     BITDEF  AF,TTATTR,16        ; return TAvailFonts

 STRUCTURE  AF,0                 ; AvailFonts
     UWORD   af_Type             ; MEMORY, DISK, or SCALED
     STRUCT  af_Attr,ta_SIZEOF   ; text attributes for font
     LABEL   af_SIZEOF

 STRUCTURE  TAF,0                ; TAvailFonts
     UWORD   taf_Type            ; MEMORY, DISK, or SCALED
     STRUCT  taf_Attr,tta_SIZEOF ; text attributes for font
     LABEL   taf_SIZEOF

 STRUCTURE  AFH,0                ; AvailFontsHeader
     UWORD   afh_NumEntries      ; number of AvailFonts elements
     LABEL   afh_AF              ; the [T]AvailFonts elements follow here

        ENDC    ; LIBRARIES_DISKFONT_I
```

libraries/dos.i

```
        IFND    LIBRARIES_DOS_I
LIBRARIES_DOS_I SET     1
**
**      $Filename: libraries/dos.i $
**      $Release: 2.04 $
**      $Revision: 36.2 $
**      $Date: 90/07/12 $
**
**      Standard C header for AmigaDOS
**
**      (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**

        IFND    DOS_DOS_I
        INCLUDE "dos/dos.i"
        ENDC

        ENDC    ; LIBRARIES_DOS_I
```

libraries/dos_lib.i

```
        IFND    LIBRARIES_DOS_LIB_I
LIBRARIES_DOS_LIB_I     SET     1
**
**      $Filename: libraries/dos_lib.i $
**      $Release: 2.04 $
**      $Revision: 36.3 $
**      $Date: 90/11/02 $
**
**      Library interface offsets for DOS library
**      Retained only for compatibility
**
**      (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**

        IFND    DOS_DOS_LIB_I
        INCLUDE "dos/dos_lib.i"
        ENDC

        ENDC    ; LIBRARIES_DOS_LIB_I
```

libraries/dosextens.i

Page 1

```
        IFND    LIBRARIES_DOSEXTENS_I
LIBRARIES_DOSEXTENS_I SET       1
**
**      $Filename: libraries/dosextens.i $
**      $Release: 2.04 $
**      $Revision: 36.2 $
**      $Date: 90/07/12 $
**
**      DOS structures not needed for the casual AmigaDOS user
**
**      (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**

        IFND    DOS_DOSEXTENS_I
        INCLUDE "dos/dosextens.i"
        ENDC

        ENDC    ; LIBRARIES_DOSEXTENS_I
```

---

libraries/expansion.i

Page 1

```
        IFND    LIBRARIES_EXPANSION_I
LIBRARIES_EXPANSION_I   SET     1
**
**      $Filename: libraries/expansion.i $
**      $Release: 2.04 $
**      $Revision: 36.6 $
**      $Date: 90/11/05 $
**
**      External definitions for expansion.library
**
**      (C) Copyright 1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**

        IFND    EXEC_TYPES_I
        INCLUDE "exec/types.i"
        ENDC    ;EXEC_TYPES_I


EXPANSIONNAME   MACRO
                dc.b    'expansion.library',0
                ENDM


;flag for the AddDosNode() call
        BITDEF  ADN,STARTPROC,0

        ENDC    ;LIBRARIES_EXPANSION_I
        IFND    LIBRARIES_EXPANSIONBASE_I
LIBRARIES_EXPANSIONBASE_I       SET     1
**
**      $Filename: libraries/expansionbase.i $
**      $Release: 2.04 $
**      $Revision: 36.17 $
**      $Date: 91/02/13 $
**
**      Definitions for the expansion library base
**
**      (C) Copyright 1987,1988,1989,1991 Commodore-Amiga, Inc.
**          All Rights Reserved
**
        IFND    EXEC_TYPES_I
        INCLUDE "exec/types.i"
        ENDC    ; EXEC_TYPES_I

        IFND    EXEC_LIBRARIES_I
        INCLUDE "exec/libraries.i"
        ENDC    ; EXEC_LIBRARIES_I

        IFND    EXEC_SEMAPHORES_I
        INCLUDE "exec/semaphores.i"
        ENDC    ; EXEC_SEMAPHORES_I

        IFND    LIBRARIES_CONFIGVARS_I
        INCLUDE "libraries/configvars.i"
        ENDC    ; LIBRARIES_CONFIGVARS_I

**
** BootNodes are scanned by dos.library at startup.  Items found on the
** list are started by dos. BootNodes are added with the AddDosNode() or
** the V36 AddBootNode() calls.
**
  STRUCTURE     BootNode,LN_SIZE
        UWORD   bn_Flags
        APTR    bn_DeviceNode
        LABEL   BootNode_SIZEOF

**
** expansion.library has functions to manipulate most of the information in
** ExpansionBase.  Direct access is not permitted.  Use FindConfigDev()
** to scan the board list.
**
  STRUCTURE     ExpansionBase,LIB_SIZE
        UBYTE   eb_Flags                                ;read only (see below)
        UBYTE   eb_Private01                            ;private
        ULONG   eb_Private02                            ;private
        ULONG   eb_Private03                            ;private
        STRUCT  eb_Private04,CurrentBinding_SIZEOF      ;private
        STRUCT  eb_Private05,LH_SIZE                    ;private
        STRUCT  eb_MountList,LH_SIZE    ; contains struct BootNode entries
        ;...                                            ;private


; error codes
EE_OK           EQU 0
EE_LASTBOARD    EQU 40  ; could not shut him up
EE_NOEXPANSION  EQU 41  ; not enough expansion mem; board shut up
EE_NOMEMORY     EQU 42  ; not enough normal memory
EE_NOBOARD      EQU 43  ; no board at that address
EE_BADMEM       EQU 44  ; tried to add a bad memory card

; Flags
        BITDEF  EB,CLOGGED,0    ; someone could not be shutup
        BITDEF  EB,SHORTMEM,1   ; ran out of expansion mem
```

```
        BITDEF  EB,BADMEM,2     ; tried to add a bad memory card
        BITDEF  EB,DOSFLAG,3    ; reserved for use by AmigaDOS
        BITDEF  EB,KICKBACK33,4 ; reserved for use by AmigaDOS
        BITDEF  EB,KICKBACK36,5 ; reserved for use by AmigaDOS
** If the following flag is set by a floppy's bootblock code, the initial
** open of the initial shell window will be delayed until the first output
** to that shell.  Otherwise the 1.3 compatible behavior applies.
        BITDEF  EB,SILENTSTART,6


        ENDC    ; LIBRARIES_EXPANSIONBASE_I
```

libraries/filehandler.i

```
        IFND    LIBRARIES_FILEHANDLER_I
LIBRARIES_FILEHANDLER_I SET     1
**
**      $Filename: libraries/filehandler.i $
**      $Release: 2.04 $
**      $Revision: 36.2 $
**      $Date: 90/07/12 $
**
**      device and file handler specific code for AmigaDOS
**
**      (C) Copyright 1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**

        IFND    DOS_FILEHANDLER_I
        INCLUDE "dos/filehandler.i"
        ENDC

        ENDC    ; LIBRARIES_FILEHANDLER_I
```

libraries/gadtools.i

```
        IFND LIBRARIES_GADTOOLS_I
LIBRARIES_GADTOOLS_I    SET     1
**
**      $Filename: libraries/gadtools.i $
**      $Release: 2.04 $
**      $Revision: 36.10 $
**      $Date: 90/11/19 $
**
**      gadtools.library definitions
**
**      (C) Copyright 1989, 1990, Commodore-Amiga, Inc.
**      All Rights Reserved.
**

*------------------------------------------------------------------------*

        IFND EXEC_TYPES_I
        INCLUDE 'exec/types.i'
        ENDC

        IFND UTILITY_TAGITEM_I
        INCLUDE 'utility/tagitem.i'
        ENDC

        IFND INTUITION_INTUITION_I
        INCLUDE 'intuition/intuition.i'
        ENDC

*------------------------------------------------------------------------*

*       The kinds (almost classes) of gadgets in the toolkit.  Use these
*       identifiers when calling CreateGadgetA()

GENERIC_KIND    EQU     0
BUTTON_KIND     EQU     1
CHECKBOX_KIND   EQU     2
INTEGER_KIND    EQU     3
LISTVIEW_KIND   EQU     4
MX_KIND         EQU     5
NUMBER_KIND     EQU     6
CYCLE_KIND      EQU     7
PALETTE_KIND    EQU     8
SCROLLER_KIND   EQU     9
* Kind number 10 is reserved
SLIDER_KIND     EQU     11
STRING_KIND     EQU     12
TEXT_KIND       EQU     13

NUM_KINDS       EQU     14

*------------------------------------------------------------------------*

*  These two definitions are obsolete, but are here for backwards
*  compatibility.  You never need to worry about these:
GADTOOLBIT      EQU     $8000

*  Use this mask to isolate the user part: *
GADTOOLMASK     EQU     ~GADTOOLBIT

*------------------------------------------------------------------------*

*  'Or' the appropriate set together for your Window IDCMPFlags: *

ARROWIDCMP      EQU     GADGETUP!GADGETDOWN!INTUITICKS!MOUSEBUTTONS

BUTTONIDCMP     EQU     GADGETUP
```

```
CHECKBOXIDCMP    EQU     GADGETUP
INTEGERIDCMP     EQU     GADGETUP
LISTVIEWIDCMP    EQU     GADGETUP!GADGETDOWN!MOUSEMOVE!ARROWIDCMP

MXIDCMP          EQU     GADGETDOWN
NUMBERIDCMP      EQU     0
CYCLEIDCMP       EQU     GADGETUP
PALETTEIDCMP     EQU     GADGETUP

*  Use ARROWIDCMP!SCROLLERIDCMP if your scrollers have arrows: *
SCROLLERIDCMP    EQU     GADGETUP!GADGETDOWN!MOUSEMOVE
SLIDERIDCMP      EQU     GADGETUP!GADGETDOWN!MOUSEMOVE
STRINGIDCMP      EQU     GADGETUP

TEXTIDCMP        EQU     0

*------------------------------------------------------------------------*

*  Typical suggested spacing between "elements": *
INTERWIDTH       EQU     8
INTERHEIGHT      EQU     4

*------------------------------------------------------------------------*

*  Generic NewGadget used by several of the gadget classes: *

    STRUCTURE NewGadget,0

        WORD    gng_LeftEdge
        WORD    gng_TopEdge     ; gadget position
        WORD    gng_Width
        WORD    gng_Height      ;  gadget size
        APTR    gng_GadgetText  ; gadget label
        APTR    gng_TextAttr    ; desired font for gadget label
        UWORD   gng_GadgetID    ; gadget ID
        ULONG   gng_Flags       ; see below
        APTR    gng_VisualInfo  ; Set to retval of GetVisualInfo()
        APTR    gng_UserData    ; gadget UserData

        LABEL   gng_SIZEOF

*   ng_Flags control certain aspects of the gadget.  The first five control
*   the placement of the descriptive text.  All larger groups supply a
*   default:

PLACETEXT_LEFT  EQU     $0001   * Right-align text on left side
PLACETEXT_RIGHT EQU     $0002   * Left-align text on right side
PLACETEXT_ABOVE EQU     $0004   * Center text above
PLACETEXT_BELOW EQU     $0008   * Center text below
PLACETEXT_IN    EQU     $0010   * Center text on

NG_HIGHLABEL    EQU     $0020   * Highlight the label

*------------------------------------------------------------------------*

* Fill out an array of these and pass that to CreateMenus():

    STRUCTURE NewMenu,0

        UBYTE   gnm_Type                ; See below
        UBYTE   gnm_Pad                 ; alignment padding
        APTR    gnm_Label               ; Menu's label
        APTR    gnm_CommKey             ; MenuItem Command Key Equiv
        UWORD   gnm_Flags               ; Menu or MenuItem flags (see note)
        LONG    gnm_MutualExclude       ; MenuItem MutualExclude word
        APTR    gnm_UserData            ; For your own use, see note
```

```

        LABEL   gnm_SIZEOF

* Each nm_Type should be one of these:
NM_TITLE        EQU     1
NM_ITEM         EQU     2
NM_SUB          EQU     3
NM_END          EQU     0

MENU_IMAGE      EQU     128
IM_ITEM         EQU     NM_ITEM!MENU_IMAGE
IM_SUB          EQU     NM_SUB!MENU_IMAGE

*  If you set your label to NM_BARLABEL, you'll get a separator bar.
NM_BARLABEL     EQU     -1


*   The nm_Flags field is used to fill out either the Menu->Flags or
*   MenuItem->Flags field.  Note that the sense of the MENUENABLED or
*   ITEMENABLED bit is inverted between this use and Intuition's use,
*   in other words, NewMenus are enabled by default.  The following
*   labels are provided to disable them:

NM_MENUDISABLED EQU     MENUENABLED
NM_ITEMDISABLED EQU     ITEMENABLED

*   The following are pre-cleared (COMMSEQ, ITEMTEXT, and HIGHxxx are set
*   later as appropriate):

NM_FLAGMASK     EQU     ~(COMMSEQ!ITEMTEXT!HIGHFLAGS)

*   You may choose among CHECKIT, MENUTOGGLE, and CHECKED.
*   Toggle-select menuitems are of type CHECKIT!MENUTOGGLE, along
*   with CHECKED if currently selected.  Mutually exclusive ones
*   are of type CHECKIT, and possibly CHECKED too.  The nm_MutualExclude
*   is a bit-wise representation of the items excluded by this one,
*   so in the simplest case (choose 1 among n), these flags would be
*   ~1, ~2, ~4, ~8, ~16, etc.  See the Intuition Menus chapter.

*   A UserData pointer can be associated with each Menu and MenuItem structure.
*   The CreateMenus() call allocates space for a UserData after each
*   Menu or MenuItem (header, item or sub-item).  You should use the
*   GTMENU_USERDATA or GTMENUITEM_USERDATA macro to extract it. */

GTMENU_USERDATA MACRO
                move.l  mu_SIZEOF(\1),\2
                ENDM

GTMENUITEM_USERDATA     MACRO
                move.l  mi_SIZEOF(\1),\2
                ENDM

*  Here is an old one for compatibility.  Do not use in new code!
MENU_USERDATA   MACRO
                move.l  mi_SIZEOF(\1),\2
                ENDM


*  These return codes can be obtained through the GTMN_ErrorCode tag:
GTMENU_TRIMMED  EQU     $00000001       ; Too many menus, items, or subitems,
                                        ; menu has been trimmed down
GTMENU_INVALID  EQU     $00000002       ; Invalid NewMenu array
GTMENU_NOMEM    EQU     $00000003       ; Out of memory


*------------------------------------------------------------------------*
```

```

*  Tags for toolkit functions:

GT_TagBase       EQU     TAG_USER+$80000 ; Begin counting tags

GTVI_NewWindow   EQU     GT_TagBase+$01  ; NewWindow struct for GetVisualInfo
GTVI_NWTags      EQU     GT_TagBase+$02  ; NWTags for GetVisualInfo

GT_Private0      EQU     GT_TagBase+$03  ; (private)

GTCB_Checked     EQU     GT_TagBase+$04  ; State of checkbox

GTLV_Top         EQU     GT_TagBase+$05  ; Top visible one in listview
GTLV_Labels      EQU     GT_TagBase+$06  ; List to display in listview
GTLV_ReadOnly    EQU     GT_TagBase+$07  ; TRUE if listview is to be read-only
GTLV_ScrollWidth         EQU     GT_TagBase+$08  ; Width of scrollbar

GTMX_Labels      EQU     GT_TagBase+$09  ; NULL-terminated array of labels
GTMX_Active      EQU     GT_TagBase+$0A  ; Active one in mx gadget

GTTX_Text        EQU     GT_TagBase+$0B  ; Text to display
GTTX_CopyText    EQU     GT_TagBase+$0C  ; Copy text label instead of referencing
   it

GTNM_Number      EQU     GT_TagBase+$0D  ; Number to display

GTCY_Labels      EQU     GT_TagBase+$0E  ; NULL-terminated array of labels
GTCY_Active      EQU     GT_TagBase+$0F  ; The active one in the cycle gad

GTPA_Depth       EQU     GT_TagBase+$10  ; Number of bitplanes in palette
GTPA_Color       EQU     GT_TagBase+$11  ; Palette color
GTPA_ColorOffset         EQU     GT_TagBase+$12  ; First color to use in palette
GTPA_IndicatorWidth      EQU     GT_TagBase+$13  ; Width of current-color indicat
   or
GTPA_IndicatorHeight     EQU     GT_TagBase+$14  ; Height of current-color indica
   tor

GTSC_Top         EQU     GT_TagBase+$15  ; Top visible in scroller
GTSC_Total       EQU     GT_TagBase+$16  ; Total in scroller area
GTSC_Visible     EQU     GT_TagBase+$17  ; Number visible in scroller
GTSC_Overlap     EQU     GT_TagBase+$18  ; Unused

* GT_TagBase+$19 through GT_TagBase+$25 are reserved

GTSL_Min         EQU     GT_TagBase+$26  ; Slider min value
GTSL_Max         EQU     GT_TagBase+$27  ; Slider max value
GTSL_Level       EQU     GT_TagBase+$28  ; Slider level
GTSL_MaxLevelLen         EQU     GT_TagBase+$29  ; Max length of printed level
GTSL_LevelFormat         EQU     GT_TagBase+$2A  ; Format string for level
GTSL_LevelPlace  EQU     GT_TagBase+$2B  ; Where level should be placed
GTSL_DispFunc    EQU     GT_TagBase+$2C  ; Callback for number calculation before
   display

GTST_String      EQU     GT_TagBase+$2D  ; String gadget's displayed string
GTST_MaxChars    EQU     GT_TagBase+$2E  ; Max length of string

GTIN_Number      EQU     GT_TagBase+$2F  ; Number in integer gadget
GTIN_MaxChars    EQU     GT_TagBase+$30  ; Max number of digits

GTMN_TextAttr    EQU     GT_TagBase+$31  ; MenuItem font TextAttr
GTMN_FrontPen    EQU     GT_TagBase+$32  ; MenuItem text pen color

GTBB_Recessed    EQU     GT_TagBase+$33  ; Make BevelBox recessed

GT_VisualInfo    EQU     GT_TagBase+$34  ; result of VisualInfo call

```

```
GTLV_ShowSelected        EQU     GT_TagBase+$35  ; show selected entry beneath li
   stview,
                                 ; set tag data = NULL for display-only, or pointer
                                 ; to a string gadget you've created
GTLV_Selected    EQU     GT_TagBase+$36  ; Set ordinal number of selected entry i
   n the list
GT_Reserved0     EQU     GT_TagBase+$37  ; Reserved
GT_Reserved1     EQU     GT_TagBase+$38  ; Reserved for future use

GTTX_Border      EQU     GT_TagBase+$39  ; Put a border around Text-display gadge
   ts
GTNM_Border      EQU     GT_TagBase+$3A  ; Put a border around Number-display gad
   gets

GTSC_Arrows      EQU     GT_TagBase+$3B  ; Specify size of arrows for scroller
GTMN_Menu        EQU     GT_TagBase+$3C  ; Pointer to Menu for use by
                                 ; LayoutMenuItems()
GTMX_Spacing     EQU     GT_TagBase+$3D  ; Added to font height to
                                 ; figure spacing between mx choices.  Use this
                                 ; instead of LAYOUTA_SPACING for mx gadgets.

*  New to V37 GadTools.  Ignored by GadTools V36.
GTMN_FullMenu    EQU     GT_TagBase+$3E  ; Asks CreateMenus() to
                 ; validate that this is a complete menu structure
GTMN_SecondaryError      EQU     GT_TagBase+$3F  ; ti_Data is a pointer
                 ; to a ULONG to receive error reports from CreateMenus()
GT_Underscore    EQU     GT_TagBase+$40  ; ti_Data points to the symbol
                 ; that preceeds the character you'd like to underline in a
                 ; gadget label

*----------------------------------------------------------------------------*

* "NWay" is an old synonym for cycle gadgets

NWAY_KIND        EQU     CYCLE_KIND
NWAYIDCMP        EQU     CYCLEIDCMP

GTNW_Labels      EQU     GTCY_Labels
GTNW_Active      EQU     GTCY_Active

         ENDC
```

```
        IFND    IFF_IFFPARSE_I
IFF_IFFPARSE_I  SET     1
**
**      $Filename: libraries/iffparse.i $
**      $Release: 2.04 $
**      $Revision: 33.1 $
**      $Date: 90/11/20 $
**
**      Assembly include file for iffparse.library.
**      Generated by hand from iffparse.h (9006.04)
**
**      (C) Copyright 1989,1990 Commodore-Amiga Inc., Stuart Ferguson
**      and Leo L. Schwab
**              All Rights Reserved
**

                IFND    EXEC_TYPES_I
                include "exec/types.i"
                ENDC

                IFND    EXEC_LISTS_I
                include "exec/lists.i"
                ENDC

                IFND    EXEC_PORTS_I
                include "exec/ports.i"
                ENDC

                IFND    DEVICES_CLIPBOARD_I
                include "devices/clipboard.i"
                ENDC

*\
* Struct associated with an active IFF stream.
* "iff_Stream" is a value used by the client's read/write/seek functions -
* it will not be accessed by the library itself and can have any value
* (could even be a pointer or a BPTR).
*/
        STRUCTURE IFFHandle,0
                ULONG   iff_Stream
                ULONG   iff_Flags
                LONG    iff_Depth       ; Depth of context stack.
* There are private fields hiding here.
        LABEL   iff_SIZEOF

*\
* Bit masks for "iff_Flags" field.
*/
IFFF_READ       EQU     0                       ; read mode - default
IFFF_WRITE      EQU     1                       ; write mode
IFFF_RWBITS     EQU     IFFF_READ!IFFF_WRITE    ; read/write bits
IFFF_FSEEK      EQU     1<<1                    ; forward seek only
IFFF_RSEEK      EQU     1<<2                    ; random seek
IFFF_RESERVED   EQU     $FFFF0000               ; Don't touch these bits.

*\
* When the library calls your stream handler, you'll be passed a pointer
* to this structure as the "message packet".
* NOTE:  ASSEMBLY PREFIX (isc_) DIFFERENT FROM C PREFIX (sc_).
*/
        STRUCTURE IFFStreamCmd,0
                LONG    isc_Command     ; Operation to be performed (IFFCMD_)
                APTR    isc_Buf         ; Pointer to data buffer
                LONG    isc_NBytes      ; Number of bytes to be affected
        LABEL   isc_SIZEOF

```

```
*\
* A node associated with a context on the iff_Stack.  Each node
* represents a chunk, the stack representing the current nesting
* of chunks in the open IFF file.  Each context node has associated
* local context items in the (private) LocalItems list.  The ID, type,
* size and scan values describe the chunk associated with this node.
*/
        STRUCTURE ContextNode,MLN_SIZE  ; cn_Node
                LONG    cn_ID
                LONG    cn_Type
                LONG    cn_Size         ; Size of this chunk
                LONG    cn_Scan         ; # of bytes read/written so far
* There are private fields hiding here.
        LABEL   cn_SIZEOF

*\
* Local context items live in the ContextNode's.  Each class is identified
* by its lci_Ident code and has a (private) purge vector for when the
* parent context node is popped.
*/
        STRUCTURE LocalContextItem,MLN_SIZE     ; lci_Node
                ULONG   lci_ID
                ULONG   lci_Type
                ULONG   lci_Ident
* There are private fields hiding here.
        LABEL   lci_SIZEOF

*\
* StoredProperty: a local context item containing the data stored
* from a previously encountered property chunk.
* NOTE:  ASSEMBLY PREFIX (spr_) DIFFERENT FROM C PREFIX (sp_).
*/
        STRUCTURE StoredProperty,0
                LONG    spr_Size
                APTR    spr_Data
        LABEL   spr_SIZEOF

*\
* Collection Item: the actual node in the collection list at which
* client will look.  The next pointers cross context boundaries so
* that the complete list is accessable.
* NOTE:  ASSEMBLY PREFIX (cit_) DIFFERENT FROM C PREFIX (ci_).
*/
        STRUCTURE CollectionItem,0
                APTR    cit_Next
                LONG    cit_Size
                APTR    cit_Data
        LABEL   cit_SIZEOF

*\
* Structure returned by OpenClipboard().  You may do CMD_POSTs and such
* using this structure.  However, once you call OpenIFF(), you may not
* do any more of your own I/O to the clipboard until you call CloseIFF().
*/
        STRUCTURE ClipboardHandle,iocr_SIZEOF   ; cbh_Reg
                STRUCT  cbh_CBport,MP_SIZE
                STRUCT  cbh_SatisfyPort,MP_SIZE
        LABEL   cbh_SIZEOF

*\
* IFF return codes.  Most functions return either zero for success or
* one of these codes.  The exceptions are the read/write functions which
* return positive values for number of bytes or records read or written,
* or a negative error code.  Some of these codes are not errors per sae,
* but valid conditions such as EOF or EOC (End of Chunk).
*/
```

libraries/iffparse.i

```
IFFERR_EOF              EQU     -1      ; Reached logical end of file
IFFERR_EOC              EQU     -2      ; About to leave context
IFFERR_NOSCOPE          EQU     -3      ; No valid scope for property
IFFERR_NOMEM            EQU     -4      ; Internal memory alloc failed
IFFERR_READ             EQU     -5      ; Stream read error
IFFERR_WRITE            EQU     -6      ; Stream write error
IFFERR_SEEK             EQU     -7      ; Stream seek error
IFFERR_MANGLED          EQU     -8      ; Data in file is corrupt
IFFERR_SYNTAX           EQU     -9      ; IFF syntax error
IFFERR_NOTIFF           EQU     -10     ; Not an IFF file
IFFERR_NOHOOK           EQU     -11     ; No call-back hook provided
IFF_RETURN2CLIENT       EQU     -12     ; Client handler normal return

*\
* Universal IFF identifiers.
*/
ID_FORM                 EQU     'FORM'
ID_LIST                 EQU     'LIST'
ID_CAT                  EQU     'CAT '
ID_PROP                 EQU     'PROP'
ID_NULL                 EQU     '    '

*\
* Ident codes for universally recognized local context items.
*/
IFFLCI_PROP             EQU     'prop'
IFFLCI_COLLECTION       EQU     'coll'
IFFLCI_ENTRYHANDLER     EQU     'enhd'
IFFLCI_EXITHANDLER      EQU     'exhd'

*\
* Control modes for ParseIFF() function.
*/
IFFPARSE_SCAN           EQU     0
IFFPARSE_STEP           EQU     1
IFFPARSE_RAWSTEP        EQU     2

*\
* Control modes for StoreLocalItem().
*/
IFFSLI_ROOT             EQU     1       ; Store in default context
IFFSLI_TOP              EQU     2       ; Store in current context
IFFSLI_PROP             EQU     3       ; Store in topmost FORM or LIST

*\
* "Flag" for writing functions.  If you pass this value in as a size
* to PushChunk() when writing a file, the parser will figure out the
* size of the chunk for you.  (Chunk sizes >= 2**31 are forbidden by the
* IFF specification, so this works.)
*/
IFFSIZE_UNKNOWN         EQU     -1

*\
* Possible call-back command values.  (Gee, it would be nice if there was an
* ENUM macro.)
*/
IFFCMD_INIT             EQU     0       ; Prepare your stream for a session
IFFCMD_CLEANUP          EQU     1       ; Terminate stream session
IFFCMD_READ             EQU     2       ; Read bytes from stream
IFFCMD_WRITE            EQU     3       ; Write bytes to stream
IFFCMD_SEEK             EQU     4       ; Seek on stream
IFFCMD_ENTRY            EQU     5       ; You just entered a new context
IFFCMD_EXIT             EQU     6       ; You're about to leave a context
IFFCMD_PURGELCI         EQU     7       ; Purge a LocalContextItem

* Backward compatibility.  Don't use these in new code.
```

libraries/iffparse.i

```
IFFSCC_INIT             EQU     IFFCMD_INIT
IFFSCC_CLEANUP          EQU     IFFCMD_CLEANUP
IFFSCC_READ             EQU     IFFCMD_READ
IFFSCC_WRITE            EQU     IFFCMD_WRITE
IFFSCC_SEEK             EQU     IFFCMD_SEEK

                ENDC    ; IFF_IFFPARSE_I
```

libraries/mathlibrary.i

```
        IFND    LIBRARIES_MATHLIBRARY_I
LIBRARIES_MATHLIBRARY_I SET     1
**
**      $Filename: libraries/mathlibrary.i $
**      $Release: 2.04 $
**      $Revision: 1.4 $
**      $Date: 90/07/13 $
**
**      Data structure returned by OpenLibrary of:
**      mathieeedoubbas.library,mathieeedoubtrans.library
**      mathieeesingbas.library,mathieeesingtrans.library
**
**
**      (C) Copyright 1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

        ifnd EXEC_LIBRARIES_I
        include "exec/libraries.i"
        endc

        STRUCTURE MathIEEEBase,0
                STRUCT  MathIEEEBase_LibNode,LIB_SIZE
                STRUCT  MathIEEEBase_reserved,18
                APTR    MathIEEEBase_TaskOpenLib        ; hook
                APTR    MathIEEEBase_TaskCloseLib       ; hook
*       This structure may be extended in the future */
        LABEL   MathIEEEBase_SIZE

;       Math resources may need to know when a program opens or closes this
;       library. The functions TaskOpenLib and TaskCloseLib are called when
;       a task opens or closes this library. The yare initialized to point
;       local initialization pertaining to 68881 stuff if 68881 resources
;       are found. To override the default the vendor must provide appropriate
;       hooks in the MathIEEEResource. If specified, these will be called
;       when the library initializes.

        ENDC    ; LIBRARIES_MATHLIBRARY_I
```

---

libraries/mathresource.i

```
        IFND    RESOURCES_MATHRESOURCE_I
RESOURCES_MATHRESOURCE_I        SET     1
**
**      $Filename: libraries/mathresource.i $
**      $Release: 2.04 $
**      $Revision: 36.3 $
**      $Date: 90/07/13 $
**
**      Data structure returned by OpenResource of:
**      "MathIEEE.resource"
**
**
**      (C) Copyright 1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

        IFND EXEC_NODES_I
        include "exec/nodes.i"
        ENDC

*
*       The 'Init' entries are only used if the corresponding
*       bit is set in the Flags field.
*
*       So if you are just a 68881, you do not need the Init stuff
*       just make sure you have cleared the Flags field.
*
*       This should allow us to add Extended Precision later.
*
*       For Init users, if you need to be called whenever a task
*       opens this library for use, you need to change the appropriate
*       entries in MathIEEELibrary.
*


        STRUCTURE MathIEEEResourceResource,0
                STRUCT  MathIEEEResource_Node,LN_SIZE
                USHORT  MathIEEEResource_Flags
                APTR    MathIEEEResource_BaseAddr       * ptr to 881 if exists *
                APTR    MathIEEEResource_DblBasInit
                APTR    MathIEEEResource_DblTransInit
                APTR    MathIEEEResource_SglBasInit
                APTR    MathIEEEResource_SglTransInit
                APTR    MathIEEEResource_ExtBasInit
                APTR    MathIEEEResource_ExtTransInit
        LABEL   MathIEEEResourceResource_SIZE

* definations for MathIEEERESOURCE_FLAGS *
        BITDEF  MATHIEEERESOURCE,DBLBAS,0
        BITDEF  MATHIEEERESOURCE,DBLTRANS,1
        BITDEF  MATHIEEERESOURCE,SGLBAS,2
        BITDEF  MATHIEEERESOURCE,SGLTRANS,3
        BITDEF  MATHIEEERESOURCE,EXTBAS,4
        BITDEF  MATHIEEERESOURCE,EXTTRANS,5

        ENDC    ; RESOURCES_MATHRESOURCE_I
```

libraries/translator.i

```
        IFND    LIBRARIES_TRANSLATOR_I
LIBRARIES_TRANSLATOR_I  SET     1
**
**      $Filename: libraries/translator.i $
**      $Release: 2.04 $
**      $Revision: 36.1 $
**      $Date: 90/12/13 $
**
**      Useful definitions for translator.library
**
**      (C) Copyright 1988, 1989, 1990 Commodore-Amiga Inc. and
**      Joseph Katz/Mark Barton.  All rights reserved.
**


*               ;------ Translator error codes
TR_NotUsed      EQU     -1              ;This is an often used system rc
TR_NoMem        EQU     -2              ;Can't allocate memory
TR_MakeBad      EQU     -4              ;Error in MakeLibrary call


        ENDC    ; LIBRARIES_TRANSLATOR_I
```

resources/battclock.i

```
        IFND    RESOURCES_BATTCLOCK_I
RESOURCES_BATTCLOCK_I   SET     1
**
**      $Filename: resources/battclock.i $
**      $Release: 2.04 $
**      $Revision: 36.3 $
**      $Date: 90/05/01 $
**
**      BattClock resource name strings.
**
**      (C) Copyright 1989,1990 Commodore-Amiga Inc.
**              All Rights Reserved
**

BATTCLOCKNAME   MACRO
                dc.b    'battclock.resource',0
                ds.w    0
                ENDM

        ENDC    ; RESOURCES_BATTCLOCK_I
```

---

resources/battmem.i

```
        IFND    RESOURCES_BATTMEM_I
RESOURCES_BATTMEM_I     SET     1
**
**      $Filename: resources/battmem.i $
**      $Release: 2.04 $
**      $Revision: 36.3 $
**      $Date: 90/05/01 $
**
**      BattMem resource name strings.
**
**      (C) Copyright 1989,1990 Commodore-Amiga Inc.
**              All Rights Reserved
**

BATTMEMNAME     MACRO
                dc.b    'battmem.resource',0
                ds.w    0
                ENDM

        ENDC    ; RESOURCES_BATTMEM_I
```

resources/battmembitsamiga.i

```
        IFND    RESOURCES_BATTMEMBITSAMIGA_I
RESOURCES_BATTMEMBITSAMIGA_I    SET     1
**
**      $Filename: resources/battmembitsamiga.i $
**      $Release: 2.04 $
**      $Revision: 1.2 $
**      $Date: 90/05/29 $
**
**      BattMem Amiga specific bit definitions.
**
**      (C) Copyright 1989,1990 Commodore-Amiga Inc.
**              All Rights Reserved
**


*
* Amiga specific bits in the battery-backedup ram.
*
*       Bits 0 to 31, inclusive
*

*
* AMIGA_AMNESIA
*
*               The battery-backedup memory has had a memory loss.
*               This bit is used as a flag that the user should be
*               notified that all battery-backed bit have been
*               reset and that some attention is required. Zero
*               indicates that a memory loss has occured.
*

BATTMEM_AMIGA_AMNESIA_ADDR      EQU     0
BATTMEM_AMIGA_AMNESIA_LEN       EQU     1


*
* SCSI_TIMEOUT
*
*               adjusts the timeout value for SCSI device selection.  A
*               value of 0 will produce short timeouts (128 ms) while a
*               value of 1 produces long timeouts (2 sec).  This is used
*               for SeaCrate drives (and some Maxtors apparently) that
*               don't respond to selection until they are fully spun up
*               and intialised.
*

BATTMEM_SCSI_TIMEOUT_ADDR       EQU     1
BATTMEM_SCSI_TIMEOUT_LEN        EQU     1


*
* SCSI_LUNS
*
*               Determines if the controller attempts to access logical
*               units above 0 at any given SCSI address.  This prevents
*               problems with drives that respond to ALL LUN addresses
*               (instead of only 0 like they should).  Default value is
*               0 meaning don't support LUNs.
*

BATTMEM_SCSI_LUNS_ADDR          EQU     2
BATTMEM_SCSI_LUNS_LEN           EQU     1


        ENDC    ; RESOURCES_BATTMEMBITSAMIGA_I
```

---

resources/battmembitsamix.i

```
        IFND    RESOURCES_BATTMEMBITSAMIX_I
RESOURCES_BATTMEMBITSAMIX_I     SET     1
**
**      $Filename: resources/battmembitsamix.i $
**      $Release: 2.04 $
**      $Revision: 1.2 $
**      $Date: 90/05/29 $
**
**      BattMem Amix specific bit definitions.
**
**      (C) Copyright 1989,1990 Commodore-Amiga Inc.
**              All Rights Reserved
**


*
*       See Amix documentation for these bit definitions
*
*       Bits 32 to 63, inclusive
*


        ENDC    ; RESOURCES_BATTMEMBITSAMIX_I
```

resources/battmembitsshared.i

```
        IFND    RESOURCES_BATTMEMSHARED_I
RESOURCES_BATTMEMBITSSHARED_I   SET     1
**
**      $Filename: resources/battmembitsshared.i $
**      $Release: 2.04 $
**      $Revision: 1.2 $
**      $Date: 90/05/29 $
**
**      BattMem shared specific bit definitions.
**
**      (C) Copyright 1989,1990 Commodore-Amiga Inc.
**              All Rights Reserved
**


*
* Shared bits in the battery-backedup ram.
*
*       Bits 64 and above
*

*
* SHARED_AMNESIA
*
*               The battery-backedup memory has had a memory loss.
*               This bit is used as a flag that the user should be
*               notified that all battery-backed bit have been
*               reset and that some attention is required. Zero
*               indicates that a memory loss has occured.
*

BATTMEM_SHARED_AMNESIA_ADDR     EQU     64
BATTMEM_SHARED_AMNESIA_LEN      EQU     1


*
* SCSI_HOST_ID
*
*               a 3 bit field (0-7) that is stored in complemented form
*               (this is so that default value of 0 really means 7)
*               It's used to set the A3000 controllers SCSI ID (on reset)
*

BATTMEM_SCSI_HOST_ID_ADDR       EQU     65
BATTMEM_SCSI_HOST_ID_LEN        EQU     3


*
* SCSI_SYNC_XFER
*
*               determines if the driver should initiate synchronous
*               transfer requests or leave it to the drive to send the
*               first request.  This supports drives that crash or
*               otherwise get confused when presented with a sync xfer
*               message.  Default=0=sync xfer not initiated.
*

BATTMEM_SCSI_SYNC_XFER_ADDR     EQU     68
BATTMEM_SCSI_SYNC_XFER_LEN      EQU     1


        ENDC    ; RESOURCES_BATTMEMSHARED_I
```

---

resources/cia.i

```
        IFND    DEVICES_CIA_I
DEVICES_CIA_I   SET     1
**
**      $Filename: resources/cia.i $
**      $Release: 2.04 $
**      $Revision: 36.3 $
**      $Date: 90/05/01 $
**
**      Cia resource name strings.
**
**      (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga Inc.
**              All Rights Reserved
**

CIAANAME        MACRO
                DC.B    'ciaa.resource',0
                ENDM

CIABNAME        MACRO
                DC.B    'ciab.resource',0
                ENDM

        ENDC    ; DEVICES_CIA_I
```

resources/ciabase.i

```
        IFND    RESOURCES_CIA_I
RESOURCES_CIA_I SET     1
**
**      $Filename: resources/ciabase.i $
**      $Release: 2.04 $
**      $Revision: 1.2 $
**      $Date: 90/05/16 $
**
**      cia base definitions
**
**      (C) Copyright 1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**


;       There is no public information in CiaBase


        ENDC    ; RESOURCES_CIA_I
```

---

resources/disk.i

```
        IFND    RESOURCES_DISK_I
RESOURCES_DISK_I        SET     1
**
**      $Filename: resources/disk.i $
**      $Release: 2.04 $
**      $Revision: 27.10 $
**      $Date: 90/11/21 $
**
**      disk.i -- external declarations for the disk resource
**
**      (C) Copyright 1985,1986,1987,1988,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**

        IFND    EXEC_TYPES_I
        INCLUDE "exec/types.i"
        ENDC    !EXEC_TYPES_I

        IFND    EXEC_LISTS_I
        INCLUDE "exec/lists.i"
        ENDC    !EXEC_LISTS_I

        IFND    EXEC_PORTS_I
        INCLUDE "exec/ports.i"
        ENDC    !EXEC_PORTS_I

        IFND    EXEC_INTERRUPTS_I
        INCLUDE "exec/interrupts.i"
        ENDC    !EXEC_INTERRUPTS_I

        IFND    EXEC_LIBRARIES_I
        INCLUDE "exec/libraries.i"
        ENDC    !EXEC_LIBRARIES_I


********************************************************************
*
* Resource structures
*
********************************************************************

    STRUCTURE DISCRESOURCEUNIT,MN_SIZE
        STRUCT  DRU_DISCBLOCK,IS_SIZE
        STRUCT  DRU_DISCSYNC,IS_SIZE
        STRUCT  DRU_INDEX,IS_SIZE
        LABEL   DRU_SIZE



    STRUCTURE DISCRESOURCE,LIB_SIZE
        APTR    DR_CURRENT          ; pointer to current unit structure
        UBYTE   DR_FLAGS
        UBYTE   DR_pad
        APTR    DR_SYSLIB
        APTR    DR_CIARESOURCE
        STRUCT  DR_UNITID,4*4
        STRUCT  DR_WAITING,LH_SIZE
        STRUCT  DR_DISCBLOCK,IS_SIZE
        STRUCT  DR_DISCSYNC,IS_SIZE
        STRUCT  DR_INDEX,IS_SIZE
        APTR    DR_CURRTASK         ; pointer to owning task for GiveUnit
        LABEL   DR_SIZE

        BITDEF  DR,ALLOC0,0         ; unit zero is allocated
        BITDEF  DR,ALLOC1,1         ; unit one is allocated
        BITDEF  DR,ALLOC2,2         ; unit two is allocated
```

resources/disk.i

```
         BITDEF   DR,ALLOC3,3        ; unit three is allocated
         BITDEF   DR,ACTIVE,7        ; is the disc currently busy?


*********************************************************************
*
* Hardware Magic
*
*********************************************************************


DSKDMAOFF         EQU      $4000     ; idle command for dsklen register


*********************************************************************
*
* Resource specific commands
*
*********************************************************************

*-- DR_NAME is a generic macro to get the name of the resource.  This
*-- way if the name is ever changed you will pick up the change
*-- automatically.
*--
*-- Normal usage would be:
*--
*-- internalName:           DISKNAME
*--

DISKNAME:         MACRO
                  DC.B     'disk.resource',0
                  DS.W     0
                  ENDM

         LIBINIT  LIB_BASE
         LIBDEF   DR_ALLOCUNIT
         LIBDEF   DR_FREEUNIT
         LIBDEF   DR_GETUNIT
         LIBDEF   DR_GIVEUNIT
         LIBDEF   DR_GETUNITID
         LIBDEF   DR_READUNITID

DR_LASTCOMM       EQU      DR_READUNITID


*********************************************************************
*
* drive types
*
*********************************************************************

DRT_AMIGA         EQU      $00000000
DRT_37422D2S      EQU      $55555555
DRT_EMPTY         EQU      $FFFFFFFF
DRT_150RPM        EQU      $AAAAAAAA

         ENDC     ; RESOURCES_DISK_I
```

resources/filesysres.i

```
         IFND     RESOURCES_FILESYSRES_I
RESOURCES_FILESYSRES_I  SET      1
**
**       $Filename: resources/filesysres.i $
**       $Release: 2.04 $
**       $Revision: 36.4 $
**       $Date: 90/05/03 $
**
**       FileSystem.resource description
**
**       (C) Copyright 1988,1989 Commodore-Amiga, Inc.
**           All Rights Reserved
**

         IFND     EXEC_NODES_I
         INCLUDE  "exec/nodes.i"
         ENDC
         IFND     EXEC_LISTS_I
         INCLUDE  "exec/lists.i"
         ENDC
         IFND     DOS_DOS_I
         INCLUDE  "dos/dos.i"
         ENDC

FSRNAME  MACRO
                  dc.b     'FileSystem.resource',0
         ENDM

 STRUCTURE  FileSysResource,LN_SIZE      ; on resource list
    CPTR    fsr_Creator                  ; name of creator of this resource
    STRUCT  fsr_FileSysEntries,LH_SIZE   ; list of FileSysEntry structs
    LABEL   FileSysResource_SIZEOF

 STRUCTURE  FileSysEntry,LN_SIZE         ; on fsr_FileSysEntries list
                                         ; LN_NAME is of creator of this entry
    ULONG   fse_DosType          ; DosType of this FileSys
    ULONG   fse_Version          ; Version of this FileSys
    ULONG   fse_PatchFlags       ; bits set for those of the following that need
                                 ;   to be substituted into a standard device
                                 ;   node for this file system: e.g. $180
                                 ;   for substitute SegList & GlobalVec
    ULONG   fse_Type             ; device node type: zero
    CPTR    fse_Task             ; standard dos "task" field
    BPTR    fse_Lock             ; not used for devices: zero
    BSTR    fse_Handler          ; filename to loadseg (if SegList is null)
    ULONG   fse_StackSize        ; stacksize to use when starting task
    LONG    fse_Priority         ; task priority when starting task
    BPTR    fse_Startup          ; startup msg: FileSysStartupMsg for disks
    BPTR    fse_SegList          ; code to run to start new task
    BPTR    fse_GlobalVec        ; BCPL global vector when starting task
    ; no more entries need exist than those implied by fse_PatchFlags

         ENDC     ; RESOURCES_FILESYSRES_I
```

resources/mathresource.i

```
        IFND    RESOURCES_MATHRESOURCE_I
RESOURCES_MATHRESOURCE_I        SET     1
**
**      $Filename: resources/mathresource.i $
**      $Release: 2.04 $
**      $Revision: 36.3 $
**      $Date: 90/07/13 $
**
**      Data structure returned by OpenResource of:
**      "MathIEEE.resource"
**
**
**      (C) Copyright 1987,1988,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

        IFND EXEC_NODES_I
        include "exec/nodes.i"
        ENDC

*
*       The 'Init' entries are only used if the corresponding
*       bit is set in the Flags field.
*
*       So if you are just a 68881, you do not need the Init stuff
*       just make sure you have cleared the Flags field.
*
*       This should allow us to add Extended Precision later.
*
*       For Init users, if you need to be called whenever a task
*       opens this library for use, you need to change the appropriate
*       entries in MathIEEELibrary.
*

        STRUCTURE MathIEEEResourceResource,0
                STRUCT  MathIEEEResource_Node,LN_SIZE
                USHORT  MathIEEEResource_Flags
                APTR    MathIEEEResource_BaseAddr       * ptr to 881 if exists *
                APTR    MathIEEEResource_DblBasInit
                APTR    MathIEEEResource_DblTransInit
                APTR    MathIEEEResource_SglBasInit
                APTR    MathIEEEResource_SglTransInit
                APTR    MathIEEEResource_ExtBasInit
                APTR    MathIEEEResource_ExtTransInit
        LABEL   MathIEEEResourceResource_SIZE

* definations for MathIEEERESOURCE_FLAGS *
        BITDEF  MATHIEEERESOURCE,DBLBAS,0
        BITDEF  MATHIEEERESOURCE,DBLTRANS,1
        BITDEF  MATHIEEERESOURCE,SGLBAS,2
        BITDEF  MATHIEEERESOURCE,SGLTRANS,3
        BITDEF  MATHIEEERESOURCE,EXTBAS,4
        BITDEF  MATHIEEERESOURCE,EXTTRANS,5

        ENDC    ; RESOURCES_MATHRESOURCE_I
```

resources/misc.i

```
        IFND    RESOURCES_MISC_I
RESOURCES_MISC_I        SET     1
**
**      $Filename: resources/misc.i $
**      $Release: 2.04 $
**      $Revision: 36.12 $
**      $Date: 90/05/06 $
**
**      Unit number definitions for "misc.resource"
**
**      (C) Copyright 1985,1989,1990 Commodore-Amiga, Inc.
**          All Rights Reserved
**

        IFND    EXEC_TYPES_I
        INCLUDE "exec/types.i"
        ENDC    !EXEC_TYPES_I

        IFND    EXEC_LIBRARIES_I
        INCLUDE "exec/libraries.i"
        ENDC    !EXEC_LIBRARIES_I

*
* Unit number definitions.  Ownership of a resource grants low-level
* bit access to the hardware registers.  You are still obligated to follow
* the rules for shared access of the interrupt system.  (see
* exec.library/SetIntVector or cia.resource as appropriate).
*
MR_SERIALPORT   EQU 0   ;Amiga custom chip serial port registers & interrupts
                        ;(SERDAT,SERDATR,SERPER,ADKCON, and interrupts)
MR_SERIALBITS   EQU 1   ;Serial control bits (DTR,CTS, etc.)
MR_PARALLELPORT EQU 2   ;The 8 bit parallel data port
                        ;(CIAAPRA & CIAADDRA only!)
MR_PARALLELBITS EQU 3   ;All other parallel bits & interrupts (BUSY,ACK,etc.)

*
* Library vector offset definitions
*
        LIBINIT LIB_BASE
        LIBDEF  MR_ALLOCMISCRESOURCE    ;-6
        LIBDEF  MR_FREEMISCRESOURCE     ;-12

*
* Name of misc.resource
*
MISCNAME        MACRO
                DC.B    'misc.resource',0
                CNOP    0,2
                ENDM

        ENDC    ;RESOURCE_MISC_I
```

resources/potgo.i

```
        IFND    RESOURCES_POTGO_I
RESOURCES_POTGO_I       EQU     1
**
**      $Filename: resources/potgo.i $
**      $Release: 2.04 $
**      $Revision: 36.0 $
**      $Date: 90/04/13 $
**
**      potgo resource name
**
**      (C) Copyright 1986,1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**
POTGONAME MACRO
                dc.b    'potgo.resource',0
                ds.w    0
        ENDM

        ENDC    ; RESOURCES_POTGO_I
```

rexx/errors.i

```
         IFND     REXX_ERRORS_I
REXX_ERRORS_I SET     1
**
**      $Filename: rexx/errors.i $
**      $Release: 2.04 $
**      $Revision: 1.4 $
**      $Date: 90/07/12 $
**
**      Definitions for ARexx error codes
**
**      (C) Copyright 1986,1987,1988,1989,1990 William S. Hawes.
**              All Rights Reserved
**

ERRC_MSG     EQU    0                  ; error code offset
ERR10_001    EQU    (ERRC_MSG+1)       ; program not found
ERR10_002    EQU    (ERRC_MSG+2)       ; execution halted
ERR10_003    EQU    (ERRC_MSG+3)       ; no memory available
                                       ; unassigned
ERR10_005    EQU    (ERRC_MSG+5)       ; unmatched quote
ERR10_006    EQU    (ERRC_MSG+6)       ; unterminated comment
                                       ; unassigned
ERR10_008    EQU    (ERRC_MSG+8)       ; unrecognized token
ERR10_009    EQU    (ERRC_MSG+9)       ; symbol or string too long

ERR10_010    EQU    (ERRC_MSG+10)      ; invalid message packet
ERR10_011    EQU    (ERRC_MSG+11)      ; command string error
ERR10_012    EQU    (ERRC_MSG+12)      ; error return from function
ERR10_013    EQU    (ERRC_MSG+13)      ; host environment not found
ERR10_014    EQU    (ERRC_MSG+14)      ; required library not available
ERR10_015    EQU    (ERRC_MSG+15)      ; function not found
ERR10_016    EQU    (ERRC_MSG+16)      ; function did not return value
ERR10_017    EQU    (ERRC_MSG+17)      ; wrong number of arguments
ERR10_018    EQU    (ERRC_MSG+18)      ; invalid argument to function
ERR10_019    EQU    (ERRC_MSG+19)      ; invalid PROCEDURE instruction

ERR10_020    EQU    (ERRC_MSG+20)      ; unexpected THEN/ELSE
ERR10_021    EQU    (ERRC_MSG+21)      ; unexpected WHEN/OTHERWISE
ERR10_022    EQU    (ERRC_MSG+22)      ; unexpected BREAK, LEAVE or ITERATE
ERR10_023    EQU    (ERRC_MSG+23)      ; invalid statement in SELECT
ERR10_024    EQU    (ERRC_MSG+24)      ; missing or multiple THEN clauses
ERR10_025    EQU    (ERRC_MSG+25)      ; missing OTHERWISE
ERR10_026    EQU    (ERRC_MSG+26)      ; missing or unexpected END
ERR10_027    EQU    (ERRC_MSG+27)      ; symbol mismatch on END/LEAVE/ITERATE
ERR10_028    EQU    (ERRC_MSG+28)      ; invalid 'DO' syntax
ERR10_029    EQU    (ERRC_MSG+29)      ; incomplete DO/IF/SELECT

ERR10_030    EQU    (ERRC_MSG+30)      ; label not found
ERR10_031    EQU    (ERRC_MSG+31)      ; symbol expected
ERR10_032    EQU    (ERRC_MSG+32)      ; string or symbol expected
ERR10_033    EQU    (ERRC_MSG+33)      ; invalid sub-keyword
ERR10_034    EQU    (ERRC_MSG+34)      ; required keyword missing
ERR10_035    EQU    (ERRC_MSG+35)      ; extraneous characters in clause
ERR10_036    EQU    (ERRC_MSG+36)      ; sub-keyword conflict
ERR10_037    EQU    (ERRC_MSG+37)      ; invalid template
                                       ; unassigned
ERR10_039    EQU    (ERRC_MSG+39)      ; uninitialized variable

ERR10_040    EQU    (ERRC_MSG+40)      ; invalid variable name
ERR10_041    EQU    (ERRC_MSG+41)      ; invalid expression
ERR10_042    EQU    (ERRC_MSG+42)      ; unbalanced parentheses
ERR10_043    EQU    (ERRC_MSG+43)      ; nesting level exceeded
ERR10_044    EQU    (ERRC_MSG+44)      ; invalid expression result
ERR10_045    EQU    (ERRC_MSG+45)      ; expression required
ERR10_046    EQU    (ERRC_MSG+46)      ; boolean value not 0 or 1
ERR10_047    EQU    (ERRC_MSG+47)      ; arithmetic conversion error
```

rexx/errors.i

```
ERR10_048    EQU    (ERRC_MSG+48)      ; invalid operand

* Return Codes for general use ...
RC_FAIL      EQU    -1                 ; something's wrong
RC_OK        EQU    0                  ; success
RC_WARN      EQU    5                  ; A warning only
RC_ERROR     EQU    10                 ; Something wrong
RC_FATAL     EQU    20                 ; Complete or severe failure

         ENDC
```

rexx/rexxio.i Page 1

```
        IFND     REXX_REXXIO_I
REXX_REXXIO_I SET     1
**
**      $Filename: rexx/rexxio.i $
**      $Release: 2.04 $
**      $Revision: 1.6 $
**      $Date: 90/11/02 $
**
**      Include file for Input/Output related structures
**
**      (C) Copyright 1986,1987,1988,1989,1990 William S. Hawes.
**              All Rights Reserved
**

        IFND     REXX_STORAGE_I
        INCLUDE  "rexx/storage.i"
        ENDC

RXBUFFSZ EQU     204                    ; buffer length

* The IoBuff is a resource node used to maintain the File List.  Nodes are
* allocated and linked into the list whenever a file is opened.

        STRUCTURE IoBuff,RRSIZEOF       ; structure for files/strings
        APTR     iobRpt                 ; read/write pointer
        LONG     iobRct                 ; character count
        LONG     iobDFH                 ; DOS filehandle
        APTR     iobLock                ; DOS lock
        LONG     iobBct                 ; buffer length
        STRUCT   iobArea,RXBUFFSZ       ; buffer area
        LABEL    iobSIZEOF              ; size: 256 bytes

IOBNAME  EQU     LN_NAME                ; logical name
IOBMODE  EQU     rr_Arg1                ; access mode
IOBEOF   EQU     rr_Arg1+1              ; EOF flag
IOBPOS   EQU     rr_Arg2                ; current position

* Access mode definitions
RXIO_EXIST  EQU     -1          ; an existing filehandle
RXIO_STRF   EQU     0                   ; a "string file"
RXIO_READ   EQU     1                   ; read-only access
RXIO_WRITE  EQU     2                   ; write mode
RXIO_APPEND EQU     3                   ; append mode (existing file)

* Offset anchors for SeekF()
RXIO_BEGIN  EQU     -1          ; relative to start
RXIO_CURR   EQU     0                   ; relative to current position
RXIO_END    EQU     1                   ; relative to end

* The Library List contains just plain resource nodes.
LLOFFSET EQU     rr_Arg1                ; "Query" offset
LLVERS   EQU     rr_Arg2                ; library version

* The RexxClipNode structure is used to maintain the Clip List.  The
* value string is stored as an argstring in the rr_Arg1 field.
CLVALUE  EQU     rr_Arg1                ; value string

* A message port structure, maintained as a resource node.
* The ReplyList holds packets that have been received but haven't been
* replied.

        STRUCTURE RexxMsgPort,RRSIZEOF
        STRUCT   rmp_Port,MP_SIZE             ; the message port
        STRUCT   rmp_ReplyList,LH_SIZE        ; messages awaiting reply
        LABEL    rmp_SIZEOF

```

rexx/rexxio.i Page 2

```
* Device types
DT_DEV   EQU      0                     ; a device
DT_DIR   EQU      1                     ; an ASSIGNed directory
DT_VOL   EQU      2                     ; a volume

* Private packet types
ACTION_STACK    EQU   2002              ; stack a line
ACTION_QUEUE    EQU   2003              ; queue a line

         ENDC
```

rexx/rxslib.i

```
         IFND     REXX_RXSLIB_I
REXX_RXSLIB_I  SET   1
**
**
**     Include file for the REXX Systems Library
**
**     (C) Copyright 1986,1987,1988,1989,1990 William S. Hawes.
**             All Rights Reserved
**

         IFND     REXX_STORAGE_I
         INCLUDE  "rexx/storage.i"
         ENDC

         ; Macro definitions

RXSLIBNAME MACRO
         dc.b     'rexxsyslib.library',0
         ENDM

RXSDIR   MACRO                      ; public port name
         dc.b     'REXX',0
         ENDM

RXSTNAME MACRO                      ; name for tasks
         dc.b     'ARexx',0
         ENDM

         ; Structure definition for the REXX systems library

         STRUCTURE RxsLib,LIB_SIZE  ; EXEC library node
         UBYTE    rl_Flags          ; global flags
         UBYTE    rl_Shadow         ; shadow flags
         APTR     rl_SysBase        ; EXEC library base
         APTR     rl_DOSBase        ; DOS library base
         APTR     rl_IeeeDPBase     ; IEEE DP math library base
         LONG     rl_SegList        ; library seglist
         LONG     rl_NIL            ; NIL: stream
         LONG     rl_Chunk          ; allocation quantum
         LONG     rl_MaxNest        ; maximum expression nesting

         APTR     rl_NULL           ; static string: NULL
         APTR     rl_FALSE          ; static string: FALSE
         APTR     rl_TRUE           ; static string: TRUE
         APTR     rl_REXX           ; static string: REXX
         APTR     rl_COMMAND        ; static string: COMMAND
         APTR     rl_STDIN          ; static string: STDIN
         APTR     rl_STDOUT         ; static string: STDOUT
         APTR     rl_STDERR         ; static string: STDERR
         APTR     rl_Version        ; version string

         APTR     rl_TaskName       ; name string for tasks
         LONG     rl_TaskPri        ; starting priority
         LONG     rl_TaskSeg        ; startup seglist
         LONG     rl_StackSize      ; stack size
         APTR     rl_RexxDir        ; REXX directory
         APTR     rl_CTABLE         ; character attribute table
         APTR     rl_Notice         ; copyright notice

         STRUCT   rl_RexxPort,MP_SIZE  ; public port
         UWORD    rl_ReadLock       ; lock count
         APTR     rl_TraceFH        ; global trace console

         STRUCT   rl_TaskList,LH_SIZE  ; REXX task list
         WORD     rl_NumTask
         STRUCT   rl_LibList,LH_SIZE   ; Library List header
```

rexx/rxslib.i

```
         WORD     rl_NumLib
         STRUCT   rl_ClipList,LH_SIZE  ; ClipList header
         WORD     rl_NumClip
         STRUCT   rl_MsgList,LH_SIZE   ; pending messages
         WORD     rl_NumMsg
         STRUCT   rl_PgmList,LH_SIZE   ; cached programs
         WORD     rl_NumPgm            ; cache count

         UWORD    rl_TraceCnt          ; trace console usage count
         WORD     rl_avail
         LABEL    rl_SIZEOF

* Global flag bit definitions for RexxMaster
RLFB_TRACE EQU    RTFB_TRACE           ; interactive tracing?
RLFB_HALT  EQU    RTFB_HALT            ; halt execution?
RLFB_SUSP  EQU    RTFB_SUSP            ; suspend execution?
RLFB_STOP  EQU    6                    ; deny further invocations
RLFB_CLOSE EQU    7                    ; close the master

* Mask for control flags
RLFMASK    EQU    1<<RLFB_TRACE!1<<RLFB_HALT!1<<RLFB_SUSP

         ; Initialization constants

RXSCHUNK EQU      1024                 ; allocation quantum
RXSNEST  EQU      32                   ; expression nesting limit
RXSTPRI  EQU      0                    ; task priority
RXSSTACK EQU      4096                 ; stack size

         ; The library entry point offsets

         LIBINIT
         LIBDEF   _LVORexx             ; Main entry point
         LIBDEF   _LVOrxParse          ; (private)
         LIBDEF   _LVOrxInstruct       ; (private)
         LIBDEF   _LVOrxSuspend        ; (private)
         LIBDEF   _LVOEvalOp           ; (private)

         LIBDEF   _LVOAssignValue      ; (private)
         LIBDEF   _LVOEnterSymbol      ; (private)
         LIBDEF   _LVOFetchValue       ; (private)
         LIBDEF   _LVOLookUpValue      ; (private)
         LIBDEF   _LVOSetValue         ; (private)
         LIBDEF   _LVOSymExpand        ; (private)

         LIBDEF   _LVOErrorMsg
         LIBDEF   _LVOIsSymbol
         LIBDEF   _LVOCurrentEnv
         LIBDEF   _LVOGetSpace
         LIBDEF   _LVOFreeSpace

         LIBDEF   _LVOCreateArgstring
         LIBDEF   _LVODeleteArgstring
         LIBDEF   _LVOLengthArgstring
         LIBDEF   _LVOCreateRexxMsg
         LIBDEF   _LVODeleteRexxMsg
         LIBDEF   _LVOClearRexxMsg
         LIBDEF   _LVOFillRexxMsg
         LIBDEF   _LVOIsRexxMsg

         LIBDEF   _LVOAddRsrcNode
         LIBDEF   _LVOFindRsrcNode
         LIBDEF   _LVORemRsrcList
         LIBDEF   _LVORemRsrcNode
         LIBDEF   _LVOOpenPublicPort
         LIBDEF   _LVOClosePublicPort
```

```
        LIBDEF  _LVOListNames

        LIBDEF  _LVOClearMem
        LIBDEF  _LVOInitList
        LIBDEF  _LVOInitPort
        LIBDEF  _LVOFreePort

        LIBDEF  _LVOCmpString
        LIBDEF  _LVOStcToken
        LIBDEF  _LVOStrcmpN
        LIBDEF  _LVOStrcmpU
        LIBDEF  _LVOStrcpyA             ; obsolete
        LIBDEF  _LVOStrcpyN
        LIBDEF  _LVOStrcpyU
        LIBDEF  _LVOStrflipN
        LIBDEF  _LVOStrlen
        LIBDEF  _LVOToUpper

        LIBDEF  _LVOCVa2i
        LIBDEF  _LVOCVi2a
        LIBDEF  _LVOCVi2arg
        LIBDEF  _LVOCVi2az
        LIBDEF  _LVOCVc2x
        LIBDEF  _LVOCVx2c

        LIBDEF  _LVOOpenF
        LIBDEF  _LVOCloseF
        LIBDEF  _LVOReadStr
        LIBDEF  _LVOReadF
        LIBDEF  _LVOWriteF
        LIBDEF  _LVOSeekF
        LIBDEF  _LVOQueueF
        LIBDEF  _LVOStackF
        LIBDEF  _LVOExistF

        LIBDEF  _LVODOSCommand
        LIBDEF  _LVODOSRead
        LIBDEF  _LVODOSWrite
        LIBDEF  _LVOCreateDOSPkt        ; obsolete
        LIBDEF  _LVODeleteDOSPkt        ; obsolete
        LIBDEF  _LVOSendDOSPkt          ; (private)
        LIBDEF  _LVOWaitDOSPkt          ; (private)
        LIBDEF  _LVOFindDevice          ; (private)

        LIBDEF  _LVOAddClipNode
        LIBDEF  _LVORemClipNode
        LIBDEF  _LVOLockRexxBase
        LIBDEF  _LVOUnlockRexxBase
        LIBDEF  _LVOCreateCLI           ; (private)
        LIBDEF  _LVODeleteCLI           ; (private)
        LIBDEF  _LVOCVs2i

* Character attribute flag bits used in REXX.
CTB_SPACE   EQU   0                     ; white space characters
CTB_DIGIT   EQU   1                     ; decimal digits 0-9
CTB_ALPHA   EQU   2                     ; alphabetic characters
CTB_REXXSYM EQU   3                     ; REXX symbol characters
CTB_REXXOPR EQU   4                     ; REXX operator characters
CTB_REXXSPC EQU   5                     ; REXX special symbols
CTB_UPPER   EQU   6                     ; UPPERCASE alphabetic
CTB_LOWER   EQU   7                     ; lowercase alphabetic

* The flag form of the character attributes
CTF_SPACE   EQU   1<<CTB_SPACE
CTF_DIGIT   EQU   1<<CTB_DIGIT
CTF_ALPHA   EQU   1<<CTB_ALPHA
```

```
CTF_REXXSYM EQU   1<<CTB_REXXSYM
CTF_REXXOPR EQU   1<<CTB_REXXOPR
CTF_REXXSPC EQU   1<<CTB_REXXSPC
CTF_UPPER   EQU   1<<CTB_UPPER
CTF_LOWER   EQU   1<<CTB_LOWER

        ENDC
```

```
         IFND REXX_STORAGE_I
REXX_STORAGE_I SET   1
**
**       $Filename: rexx/storage.i $
**       $Release: 2.04 $
**       $Revision: 1.6 $
**       $Date: 90/11/02 $
**
**       Include file for REXX data structures and memory/storage management.
**
**       (C) Copyright 1986,1987,1988,1989,1990 William S. Hawes.
**               All Rights Reserved
**

         IFND EXEC_TYPES_I
         INCLUDE "exec/types.i"
         ENDC

         IFND EXEC_NODES_I
         INCLUDE "exec/nodes.i"
         ENDC

         IFND EXEC_LISTS_I
         INCLUDE "exec/lists.i"
         ENDC

         IFND EXEC_PORTS_I
         INCLUDE "exec/ports.i"
         ENDC

         IFND EXEC_LIBRARIES_I
         INCLUDE "exec/libraries.i"
         ENDC

* The NexxStr structure is used to maintain the internal strings in REXX.
* It includes the buffer area for the string and associated attributes.
* This is actually a variable-length structure; it is allocated for a
* specific length string, and the length is never modified thereafter
* (since it's used for recycling).

         STRUCTURE NexxStr,0
         LONG    ns_Ivalue           ; integer value
         UWORD   ns_Length           ; length in bytes (excl null byte)
         UBYTE   ns_Flags            ; attribute flags
         UBYTE   ns_Hash             ; sum-of-characters hash code
         STRUCT  ns_Buff,8           ; buffer area for strings
         LABEL   NSMINSIZE           ; 16 bytes (minimum)

NXADDLEN EQU     ns_Buff+1           ; structure offset plus null byte
IVALUE   EQU     ns_Ivalue           ; integer value

* String attribute flag bit definitions
NSB_KEEP     EQU   0                 ; permanent string? (in Symbol Table)
NSB_STRING   EQU   1                 ; string form valid?
NSB_NOTNUM   EQU   2                 ; non-numeric?
NSB_NUMBER   EQU   3                 ; a valid number?
NSB_BINARY   EQU   4                 ; integer value saved?
NSB_FLOAT    EQU   5                 ; floating point format?
NSB_EXT      EQU   6                 ; an external string?
NSB_SOURCE   EQU   7                 ; part of the program source?

* The flag form of the string attributes
NSF_KEEP     EQU   1<<NSB_KEEP
NSF_STRING   EQU   1<<NSB_STRING
NSF_NOTNUM   EQU   1<<NSB_NOTNUM
NSF_NUMBER   EQU   1<<NSB_NUMBER
```

```
NSF_BINARY   EQU   1<<NSB_BINARY
NSF_FLOAT    EQU   1<<NSB_FLOAT
NSF_EXT      EQU   1<<NSB_EXT
NSF_SOURCE   EQU   1<<NSB_SOURCE

* Combinations of flags
NSF_INTNUM   EQU   (NSF_NUMBER!NSF_BINARY!NSF_STRING)
NSF_DPNUM    EQU   (NSF_NUMBER!NSF_FLOAT)
NSF_ALPHA    EQU   (NSF_NOTNUM!NSF_STRING)
NSF_OWNED    EQU   (NSF_SOURCE!NSF_EXT!NSF_KEEP)
KEEPSTR      EQU   (NSF_STRING!NSF_SOURCE!NSF_NOTNUM)
KEEPNUM      EQU   (NSF_STRING!NSF_SOURCE!NSF_NUMBER!NSF_BINARY)

* The RexxArg structure is identical to the NexxStr structure, but
* is allocated from system memory rather than from internal storage.
* This structure is used for passing arguments to external programs.
* It is usually passed as an "argstring", a pointer to the string buffer.

         STRUCTURE RexxArg,0
         LONG    ra_Size             ; total allocated length
         UWORD   ra_Length           ; length of string
         UBYTE   ra_Flags            ; attribute flags
         UBYTE   ra_Hash             ; hash code
         STRUCT  ra_Buff,8           ; buffer area
         LABEL   RexxArg_SIZEOF      ; size: 16 bytes
; Changed to RexxArg_SIZEOF from ra_SIZEOF

* The RexxMsg structure is used for all communications with Rexx programs.
* It is an EXEC message with a parameter block appended.

         STRUCTURE RexxMsg,MN_SIZE
         APTR    rm_TaskBlock        ; pointer to RexxTask structure
         APTR    rm_LibBase          ; library pointer
         LONG    rm_Action           ; command (action) code
         LONG    rm_Result1          ; return code
         LONG    rm_Result2          ; secondary result
         STRUCT  rm_Args,16*4        ; argument block (ARG0-ARG15)
         APTR    rm_PassPort         ; forwarding port
         APTR    rm_CommAddr         ; host address (port name)
         APTR    rm_FileExt          ; file extension
         LONG    rm_Stdin            ; input stream
         LONG    rm_Stdout           ; output stream
         LONG    rm_avail            ; future expansion
         LABEL   RMSIZEOF            ; size: 128 bytes
; Ranamed from rm_SIZEOF

* Field definitions
ACTION   EQU     rm_Action           ; action code
RESULT1  EQU     rm_Result1          ; primary return/error level
RESULT2  EQU     rm_Result2          ; secondary return/result string
ARG0     EQU     rm_Args             ; start of argblock
ARG1     EQU     rm_Args+4           ; first argument
ARG2     EQU     rm_Args+8           ; second argument

MAXRMARG EQU     15                  ; maximum arguments

* Command (action) codes for message packets
RXCOMM   EQU     $01000000           ; a command-level invocation
RXFUNC   EQU     $02000000           ; a function call
RXCLOSE  EQU     $03000000           ; close the port
RXQUERY  EQU     $04000000           ; query for information
RXADDFH  EQU     $07000000           ; add a function host
RXADDLIB EQU     $08000000           ; add a function library
RXREMLIB EQU     $09000000           ; remove a function library
RXADDCON EQU     $0A000000           ; add/update a ClipList string
RXREMCON EQU     $0B000000           ; remove a ClipList string
```

```
RXTCOPN  EQU      $0C000000             ; open the trace console
RXTCCLS  EQU      $0D000000             ; close the trace console

* Command modifier flag bits
RXFB_NOIO    EQU   16                   ; suppress I/O inheritance?
RXFB_RESULT EQU   17                    ; result string expected?
RXFB_STRING EQU   18                    ; program is a "string file"?
RXFB_TOKEN   EQU   19                   ; tokenize the command line?
RXFB_NONRET EQU   20                    ; a "no-return" message?

* Modifier flags
RXFF_RESULT EQU   1<<RXFB_RESULT
RXFF_STRING EQU   1<<RXFB_STRING
RXFF_TOKEN   EQU   1<<RXFB_TOKEN
RXFF_NONRET EQU   1<<RXFB_NONRET

RXCODEMASK   EQU   $FF000000
RXARGMASK    EQU   $0000000F

* The RexxRsrc structure is used to manage global resources.
* The name string for each node is created as a RexxArg structure,
* and the total size of the node is saved in the "rr_Size" field.
* Functions are provided to allocate and release resource nodes.
* If special deletion operations are required, an offset and base can
* be provided in "rr_Func" and "rr_Base", respectively.  This function
* will be called with the base in register A6 and the node in A0.

         STRUCTURE RexxRsrc,LN_SIZE
         WORD      rr_Func              ; "auto-delete" offset
         APTR      rr_Base              ; "auto-delete" base
         LONG      rr_Size              ; total size of node
         LONG      rr_Arg1              ; available ...
         LONG      rr_Arg2              ; available ...
         LABEL     RRSIZEOF             ; size: 32 bytes
; Changed from rr_SIZEOF to RRSIZEOF

* Field definitions
RRTYPE   EQU      LN_TYPE               ; node type
RRNAME   EQU      LN_NAME               ; node name (argstring)
RRSIZE   EQU      rr_Size               ; total size of node

* Resource node types
RRT_ANY  EQU      0                     ; any node type ...
RRT_LIB  EQU      1                     ; a function library
RRT_PORT EQU      2                     ; a public port
RRT_FILE EQU      3                     ; a file IoBuff
RRT_HOST EQU      4                     ; a function host
RRT_CLIP EQU      5                     ; a Clip List node

* The RexxTask structure holds the fields used by REXX to communicate with
* external processes, including the client task.  It includes the global
* data structure (and the base environment).  The structure is passed to
* the newly-created task in its "wake-up" message.

GLOBALSZ EQU      200                   ; space for the Global Data structure

         STRUCTURE RexxTask,GLOBALSZ    ; global data structure
         STRUCT    rt_MsgPort,MP_SIZE   ; message port
         UBYTE     rt_Flags             ; task flag bits
         BYTE      rt_SigBit            ; signal bit

         APTR      rt_ClientID          ; the client's task ID
         APTR      rt_MsgPkt            ; the packet being processed
         APTR      rt_TaskID            ; our task ID
         APTR      rt_RexxPort          ; the REXX public port

```

```
         APTR      rt_ErrTrap           ; Error trap address
         APTR      rt_StackPtr          ; stack pointer for traps

         STRUCT    rt_Header1,LH_SIZE
         STRUCT    rt_Header2,LH_SIZE
         STRUCT    rt_Header3,LH_SIZE
         STRUCT    rt_Header4,LH_SIZE
         STRUCT    rt_Header5,LH_SIZE
         LABEL     rt_SIZEOF

ENVLIST  EQU      rt_Header1            ; environment list (internal)
FREELIST EQU      rt_Header2            ; freelist (internal)
MEMLIST  EQU      rt_Header3            ; allocation list (external)
FILELIST EQU      rt_Header4            ; I/O files list (external)
PORTLIST EQU      rt_Header5            ; message ports list (external)
NUMLISTS EQU      5

* Definitions for RexxTask flag bits
RTFB_TRACE  EQU      0                  ; external trace flag
RTFB_HALT   EQU      1                  ; external halt flag
RTFB_SUSP   EQU      2                  ; suspend task?
RTFB_TCUSE  EQU      3                  ; trace console in use?
RTFB_WAIT   EQU      6                  ; waiting for reply?
RTFB_CLOSE  EQU      7                  ; task completed?

* Definitions for memory allocation constants
MEMQUANT EQU      16                    ; quantum of memory space
MEMMASK  EQU      $FFFFFFF0             ; mask for rounding the size

MEMQUICK EQU      (1<<0)                ; EXEC flags: MEMF_PUBLIC
MEMCLEAR EQU      (1<<16)               ; EXEC flags: MEMF_CLEAR

* The SrcNode is a temporary structure used to hold values destined for a
* segment array.  It is also used to maintain the memory freelist.

         STRUCTURE SrcNode,0            ; temporary source data structure
         APTR      sn_Succ
         APTR      sn_Pred
         APTR      sn_Ptr               ; pointer value
         LONG      sn_Size              ; size of object
         LABEL     sn_SIZEOF            ; size: 16 bytes

         ENDC
```

utility/date.i

```
        IFND    UTILITY_DATE_I
UTILITY_DATE_I  SET     1
**
**      $Filename: utility/date.i $
**      $Release: 2.04 $
**      $Revision: 36.2 $
**      $Date: 91/03/04 $
**
**      Date conversion routines ClockData definition.
**
**      (C) Copyright 1989,1990 Commodore-Amiga Inc.
**              All Rights Reserved
**

        IFND EXEC_TYPES_I
        INCLUDE "exec/types.i"
        ENDC

 STRUCTURE CLOCKDATA,0
        UWORD   sec
        UWORD   min
        UWORD   hour
        UWORD   mday
        UWORD   month
        UWORD   year
        UWORD   wday
        LABEL   CD_SIZE

        ENDC    ; UTILITY_DATE_I
```

---

utility/hooks.i

```
    IFND UTILITY_HOOKS_I
UTILITY_HOOKS_I SET 1
**
**      $Filename: utility/hooks.i $
**      $Release: 2.04 $
**      $Revision: 36.1 $
**      $Date: 90/07/12 $
**
**      callback hooks
**
**      (C) Copyright 1989,1990 Commodore-Amiga Inc.
**              All Rights Reserved
**

    IFND EXEC_TYPES_I
    INCLUDE "exec/types.i"
    ENDC

    IFND EXEC_NODES_I
    INCLUDE "exec/nodes.i"
    ENDC


; New Hook conventions
; A0 - pointer to hook itself
; A1 - pointer to parameter packed ("message")
; A2 - Hook specific address data ("object," e.g, gadget )

 STRUCTURE HOOK,MLN_SIZE
    APTR        h_Entry                 ; assembler entry point
    APTR        h_SubEntry              ; optional HLL entry point
    APTR        h_Data                  ; owner specific
 LABEL          h_SIZEOF


 ENDC
```

utility/tagitem.i

Page 1

```
    IFND UTILITY_TAGITEM_I
UTILITY_TAGITEM_I SET   1
**
**      $Filename: utility/tagitem.i $
**      $Release: 2.04 $
**      $Revision: 36.3 $
**      $Date: 91/01/24 $
**
**      extended specification mechanism
**
**      (C) Copyright 1989,1990 Commodore-Amiga Inc.
**              All Rights Reserved
**

        IFND EXEC_TYPES_I
        INCLUDE "exec/types.i"
        ENDC

; ======================================================================
; ====  TagItem ========================================================
; ======================================================================
; This data type may propagate through the system for more general use.
; In the meantime, it is used as a general mechanism of extensible data
; arrays for parameter specification and property inquiry (coming soon
; to a display controller near you).
;
; In practice, an array (or chain of arrays) of TagItems is used.

 STRUCTURE      TagItem,0
        ULONG   ti_Tag          ; identifies the type of this item
        ULONG   ti_Data         ; type-specific data, can be a pointer
        LABEL   ti_SIZEOF

; ----  system tag values ----------------------------
TAG_DONE        EQU     0  ; terminates array of TagItems. ti_Data unused
TAG_IGNORE      EQU     1  ; ignore this item, not end of array
TAG_MORE        EQU     2  ; ti_Data is pointer to another array of TagItems
                           ; note that this tag terminates the current array
TAG_SKIP        EQU     3  ; skip this and the next ti_Data items

; ----  user tag identification -----------------------
TAG_USER        EQU     $80000000  ; differentiates user tags from system tags

; until further notice, tag bits 16-30 are RESERVED and should be zero.
; Also, the value (TAG_USER | 0) should never be used as a tag value.

        ENDC
```

utility/utility.i

Page 1

```
        IFND    UTILITY_UTILITY_I
UTILITY_UTILITY_I SET 1
**
**      $Filename: utility/utility.i $
**      $Release: 2.04 $
**      $Revision: 36.1 $
**      $Date: 90/07/12 $
**
**      utility.library include file
**
**      (C) Copyright 1989 Commodore-Amiga, Inc.
**          All Rights Reserved
**

UTILITYNAME MACRO
                DC.B    'utility.library',0
        ENDM

        ENDC    ; UTILITY_UTILITY_I
```

workbench/icon.i

Page 1

```
        IFND    WORKBENCH_ICON_I
WORKBENCH_ICON_I        SET     1
**
**      $Filename: workbench/icon.i $
**      $Release: 2.04 $
**      $Revision: 36.1 $
**      $Date: 90/10/26 $
**
**      external declarations for icon.library
**
**      (C) Copyright 1985,1986,1987,1988,1989,1990, Commodore-Amiga, Inc.
**      All Rights Reserved
**

ICONNAME        MACRO
                DC.B    'icon.library',0
                ENDM

        ENDC    ; WORKBENCH_ICON_I
```

workbench/startup.i

Page 1

```
        IFND    WORKBENCH_STARTUP_I
WORKBENCH_STARTUP_I     EQU     1
**
**      $Filename: workbench/startup.i $
**      $Release: 2.04 $
**      $Revision: 36.4 $
**      $Date: 90/12/02 $
**
**      workbench startup definitions
**
**      (C) Copyright 1985,1986,1987,1988,1989,1990, Commodore-Amiga, Inc.
**      All Rights Reserved
**

        IFND    EXEC_TYPES_I
        INCLUDE "exec/types.i"
        ENDC

        IFND    EXEC_PORTS_I
        INCLUDE "exec/ports.i"
        ENDC

        IFND    LIBRARIES_DOS_I
        INCLUDE "libraries/dos.i"
        ENDC

 STRUCTURE WBStartup,0
        STRUCT  sm_Message,MN_SIZE      ; a standard message structure
        APTR    sm_Process              ; the process descriptor for you
        BPTR    sm_Segment              ; a descriptor for your code
        LONG    sm_NumArgs              ; the number of elements in ArgList
        APTR    sm_ToolWindow           ; description of window
        APTR    sm_ArgList              ; the arguments themselves
        LABEL   sm_SIZEOF

 STRUCTURE WBArg,0
        BPTR    wa_Lock                 ; a lock descriptor
        APTR    wa_Name                 ; a string relative to that lock
        LABEL   wa_SIZEOF

        ENDC    ; WORKBENCH_STARTUP_I
        IFND    WORKBENCH_WORKBENCH_I
WORKBENCH_WORKBENCH_I   EQU     1
**
**      $Filename: workbench/workbench.i $
**      $Release: 2.04 $
**      $Revision: 37.2 $
**      $Date: 91/02/01 $
**
**      workbench.library general definitions
**
**      (C) Copyright 1985,1986,1987,1988,1989,1990, Commodore-Amiga, Inc.
**      All Rights Reserved
**

        IFND    EXEC_TYPES_I
        INCLUDE "exec/types.i"
        ENDC

        IFND    EXEC_NODES_I
        INCLUDE "exec/nodes.i"
        ENDC

        IFND    EXEC_LISTS_I
        INCLUDE "exec/lists.i"
        ENDC

        IFND    EXEC_TASKS_I
        INCLUDE "exec/tasks.i"
        ENDC

        IFND    INTUITION_INTUITION_I
        INCLUDE "intuition/intuition.i"
        ENDC


; the Workbench object types
WBDISK          EQU     1
WBDRAWER        EQU     2
WBTOOL          EQU     3
WBPROJECT       EQU     4
WBGARBAGE       EQU     5
WBDEVICE        EQU     6
WBKICK          EQU     7
WBAPPICON       EQU     8

; the main workbench object structure
 STRUCTURE DrawerData,0
    STRUCT      dd_NewWindow,nw_SIZE    ; args to open window
    LONG        dd_CurrentX             ; current x coordinate of origin
    LONG        dd_CurrentY             ; current y coordinate of origin
    LABEL       OldDrawerData_SIZEOF    ; pre V36 size
; the amount of OldDrawerData actually written to disk
OLDDRAWERDATAFILESIZE   EQU (OldDrawerData_SIZEOF)
    ULONG       dd_Flags                ; flags for drawer
    UWORD       dd_ViewModes            ; view mode for drawer
    LABEL       DrawerData_SIZEOF
; the amount of DrawerData actually written to disk
DRAWERDATAFILESIZE      EQU (DrawerData_SIZEOF)


 STRUCTURE DiskObject,0
    UWORD       do_Magic                ; a magic num at the start of the file
    UWORD       do_Version              ; a version number, so we can change it
    STRUCT      do_Gadget,gg_SIZEOF     ; a copy of in core gadget
    UBYTE       do_Type
    UBYTE       do_PAD_BYTE             ; Pad it out to the next word boundry
```

```
    APTR        do_DefaultTool
    APTR        do_ToolTypes
    LONG        do_CurrentX
    LONG        do_CurrentY
    APTR        do_DrawerData
    APTR        do_ToolWindow           ; only applies to tools
    LONG        do_StackSize            ; only applies to tools
    LABEL       do_SIZEOF

WB_DISKMAGIC    EQU     $e310   ; a magic number, not easily impersonated
WB_DISKVERSION  EQU     1       ; our current version number
WB_DISKREVISION EQU     1       ; out current revision number
; I only use the lower 8 bits of Gadget.UserData for the revision #
WB_DISKREVISIONMASK     EQU     $ff

 STRUCTURE FreeList,0
    WORD        fl_NumFree
    STRUCT      fl_MemList,LH_SIZE
    ; weird name to avoid conflicts with FileLocks
    LABEL       FreeList_SIZEOF



* each message that comes into the WorkBenchPort must have a type field
* in the preceeding short.  These are the defines for this type
*

MTYPE_PSTD              EQU     1       ; a "standard Potion" message
MTYPE_TOOLEXIT          EQU     2       ; exit message from our tools
MTYPE_DISKCHANGE        EQU     3       ; dos telling us of a disk change
MTYPE_TIMER             EQU     4       ; we got a timer tick
MTYPE_CLOSEDOWN         EQU     5       ; <unimplemented>
MTYPE_IOPROC            EQU     6       ; <unimplemented>
MTYPE_APPWINDOW         EQU     7       ; msg from an app window
MTYPE_APPICON           EQU     8       ; msg from an app icon
MTYPE_APPMENUITEM       EQU     9       ; msg from an app menuitem
MTYPE_COPYEXIT          EQU     10      ; exit message from copy process
MTYPE_ICONPUT           EQU     11      ; msg from PutDiskObject in icon.library

* workbench does different complement modes for its gadgets.
* It supports separate images, complement mode, and backfill mode.
* The first two are identical to intuitions GADGIMAGE and GADGHCOMP.
* backfill is similar to GADGHCOMP, but the region outside of the
* image (which normally would be color three when complemented)
* is flood-filled to color zero.
*
GADGBACKFILL            EQU     $0001

* if an icon does not really live anywhere, set its current position
* to here
*
NO_ICON_POSITION        EQU     ($80000000)


* workbench now is a library.  this is it's name
WORKBENCH_NAME  MACRO
                dc.b            'workbench.library',0
                ds.w            0
                ENDM

; If you find am_Version >= AM_VERSION, you now this structure has
; at least the fields defined in this version of the include file
AM_VERSION      EQU     1

 STRUCTURE AppMessage,0
        STRUCT am_Message,MN_SIZE       ; standard message structure
```

workbench/workbench.i Page 3

```
        UWORD am_Type                   ; message type
        ULONG am_UserData               ; application specific
        ULONG am_ID                     ; application definable ID
        LONG am_NumArgs                 ; # of elements in arglist
        APTR am_ArgList                 ; the arguements themselves
        UWORD am_Version                ; will be AM_VERSION
        UWORD am_Class                  ; message class
        WORD am_MouseX                  ; mouse x position of event
        WORD am_MouseY                  ; mount y position of event
        ULONG am_Seconds                ; current system clock time
        ULONG am_Micros                 ; current system clock time
        STRUCT am_Reserved,8            ; avoid recompilation
        LABEL AppMessage_SIZEOF


*
* The following structures are private.  These are just stub
* structures for code compatibility...
*
 STRUCTURE AppWindow,0
        STRUCT aw_PRIVATE,0
        LABEL AppWindow_SIZEOF

 STRUCTURE AppIcon,0
        STRUCT ai_PRIVATE,0
        LABEL AppIcon_SIZEOF

 STRUCTURE AppMenuItem,0
        STRUCT ami_PRIVATE,0
        LABEL AppMenuItem_SIZEOF


        ENDC    ; WORKBENCH_WORKBENCH_I

```

Reference charts

This section contains several handy reference charts. These are often useful when searching memory or scanning structures during debugging. The charts are:

- ☐ 2.04 Function Offsets - The Amiga libraries are listed, with a separate entry for each library function. The chart lists the function's negative offset from the library base and a short summary of register usage.
- ☐ 2.04 Structure Reference - All structures in the Amiga system include files are listed alphabetically along with their sizes and the offset of each field within the structure given in hex and decimal. An exclamation point in front of the entry indicates a longword field which is not longword-aligned.
- ☐ C Language Cross Reference - Each element from the Amiga include files is listed along with its resolved value, the location where it was defined (indicated by *), and each place that references it. Since the elements in the assembly language include files have similar names, this chart is also useful for assembly language programmers.
- ☐ Amiga Function Index - This is an alphabetical listing of all the functions, device commands, and macros covered in this book showing the page number where you can find each one. Also listed is the name of the library, device, or resource where the function is located.

Function Offset Reference

Page 1

```
***************** asl
* "asl.library"
##base _AslBase
##bias 30
##public
*--- functions in V36 or higher (distributed as Release 2.0) ---
*
 30 $ffe2 -$001e AllocFileRequest()()
 36 $ffdc -$0024 FreeFileRequest(fileReq)(a0)
 42 $ffd6 -$002a RequestFile(fileReq)(a0)
 48 $ffd0 -$0030 AllocAslRequest(type,tagList)(d0/a0)
 54 $ffca -$0036 FreeAslRequest(request)(a0)
 60 $ffc4 -$003c AslRequest(request,tagList)(a0/a1)
##end

***************** battclock
* "battclock.resource"
##base _BattClockBase
##bias 6
##public
  6 $fffa -$0006 ResetBattClock()()
 12 $fff4 -$000c ReadBattClock()()
 18 $ffee -$0012 WriteBattClock(time)(d0)
##private
 24 $ffe8 -$0018 ReadBattClockMem(offset,length)(d1/d2)
 30 $ffe2 -$001e WriteBattClockMem(data,offset,length)(d0/d1/d2)
##end

***************** battmem
* "battmem.resource"
##base _BattMemBase
##bias 6
##public
  6 $fffa -$0006 ObtainBattSemaphore()()
 12 $fff4 -$000c ReleaseBattSemaphore()()
 18 $ffee -$0012 ReadBattMem(buffer,offset,length)(a0,d0/d1)
 24 $ffe8 -$0018 WriteBattMem(buffer,offset,length)(a0,d0/d1)
##end

***************** cia
* "CiaA.Resource" and "CiaB.Resource"
##bias 6
##public
  6 $fffa -$0006 AddICRVector(resource,iCRBit,interrupt)(a6,d0/a1)
 12 $fff4 -$000c RemICRVector(resource,iCRBit,interrupt)(a6,d0/a1)
 18 $ffee -$0012 AbleICR(resource,mask)(a6,d0)
 24 $ffe8 -$0018 SetICR(resource,mask)(a6,d0)
##end

***************** commodities
* "commodities.library"
##base _CxBase
##bias 30
##public
*--- functions in V36 or higher (distributed as Release 2.0) ---
*
*  OBJECT UTILITIES
*
 30 $ffe2 -$001e CreateCxObj(type,arg1,arg2)(d0/a0/a1)
 36 $ffdc -$0024 CxBroker(nb,error)(a0,d0)
 42 $ffd6 -$002a ActivateCxObj(co,true)(a0,d0)
 48 $ffd0 -$0030 DeleteCxObj(co)(a0)
 54 $ffca -$0036 DeleteCxObjAll(co)(a0)
 60 $ffc4 -$003c CxObjType(co)(a0)
 66 $ffbe -$0042 CxObjError(co)(a0)
```

Function Offset Reference

Page 2

```
 72 $ffb8 -$0048 ClearCxObjError(co)(a0)
 78 $ffb2 -$004e SetCxObjPri(co,pri)(a0,d0)
*
*  OBJECT ATTACHMENT
*
 84 $ffac -$0054 AttachCxObj(headobj,co)(a0/a1)
 90 $ffa6 -$005a EnqueueCxObj(headobj,co)(a0/a1)
 96 $ffa0 -$0060 InsertCxObj(headobj,co,pred)(a0/a1/a2)
102 $ff9a -$0066 RemoveCxObj(co)(a0)
*
*  TYPE SPECIFIC
*
##private
108 $ff94 -$006c FindBroker(name)(a0)
##public
114 $ff8e -$0072 SetTranslate(translator,ie)(a0/a1)
120 $ff88 -$0078 SetFilter(filter,text)(a0/a1)
126 $ff82 -$007e SetFilterIX(filter,ix)(a0/a1)
132 $ff7c -$0084 ParseIX(descr,ix)(a0/a1)
*
*  COMMON MESSAGE
*
138 $ff76 -$008a CxMsgType(cxm)(a0)
144 $ff70 -$0090 CxMsgData(cxm)(a0)
150 $ff6a -$0096 CxMsgID(cxm)(a0)
*
*  MESSAGE ROUTING
*
156 $ff64 -$009c DivertCxMsg(cxm,headobj,ret)(a0/a1/a2)
162 $ff5e -$00a2 RouteCxMsg(cxm,co)(a0/a1)
168 $ff58 -$00a8 DisposeCxMsg(cxm)(a0)
*
*  INPUT EVENT HANDLING
*
174 $ff52 -$00ae InvertKeyMap(ansicode,ie,km)(d0/a0/a1)
180 $ff4c -$00b4 AddIEvents(ie)(a0)
*
* FOR USE ONLY BY CONTROLLER PROGRAM
*
##private
186 $ff46 -$00ba CopyBrokerList(blist)(a0)
192 $ff40 -$00c0 FreeBrokerList(list)(a0)
198 $ff3a -$00c6 BrokerCommand(name,id)(a0,d0)
##public
##end

***************** console
* "console.device"
##base _ConsoleDevice
##bias 42
##public
 42 $ffd6 -$002a CDInputHandler(events,consoleDevice)(a0/a1)
 48 $ffd0 -$0030 RawKeyConvert(events,buffer,length,keyMap)(a0/a1,d1/a2)
##private
*--- functions in V36 or higher (distributed as Release 2.0) ---
 54 $ffca -$0036 GetConSnip()()
 60 $ffc4 -$003c SetConSnip(snip)(a0)
 66 $ffbe -$0042 AddConSnipHook(hook)(a0)
 72 $ffb8 -$0048 RemConSnipHook(hook)(a0)
##end

***************** disk
* "disk.resource"
##base _DiskBase
##bias 6
##public
```

Function Offset Reference — Page 3

```
  6 $fffa -$0006 AllocUnit(unitNum)(d0)
 12 $fff4 -$000c FreeUnit(unitNum)(d0)
 18 $ffee -$0012 GetUnit(unitPointer)(a1)
 24 $ffe8 -$0018 GiveUnit()()
 30 $ffe2 -$001e GetUnitID(unitNum)(d0)
*------ new for V37 ------
 36 $ffdc -$0024 ReadUnitID(unitNum)(d0)
##end

****************** diskfont
* "diskfont.library"
##base _DiskfontBase
##bias 30
##public
 30 $ffe2 -$001e OpenDiskFont(textAttr)(a0)
 36 $ffdc -$0024 AvailFonts(buffer,bufBytes,flags)(a0,d0/d1)
*--- functions in V34 or higher (distributed as Release 1.3) ---
 42 $ffd6 -$002a NewFontContents(fontsLock,fontName)(a0/a1)
 48 $ffd0 -$0030 DisposeFontContents(fontContentsHeader)(a1)
*--- functions in V36 or higher (distributed as Release 2.0) ---
 54 $ffca -$0036 NewScaledDiskFont(sourceFont,destTextAttr)(a0/a1)
##end

****************** dos
* "dos.library"
##base _DOSBase
##bias 30
##public
 30 $ffe2 -$001e Open(name,accessMode)(d1/d2)
 36 $ffdc -$0024 Close(file)(d1)
 42 $ffd6 -$002a Read(file,buffer,length)(d1/d2/d3)
 48 $ffd0 -$0030 Write(file,buffer,length)(d1/d2/d3)
 54 $ffca -$0036 Input()()
 60 $ffc4 -$003c Output()()
 66 $ffbe -$0042 Seek(file,position,offset)(d1/d2/d3)
 72 $ffb8 -$0048 DeleteFile(name)(d1)
 78 $ffb2 -$004e Rename(oldName,newName)(d1/d2)
 84 $ffac -$0054 Lock(name,type)(d1/d2)
 90 $ffa6 -$005a UnLock(lock)(d1)
 96 $ffa0 -$0060 DupLock(lock)(d1)
102 $ff9a -$0066 Examine(lock,fileInfoBlock)(d1/d2)
108 $ff94 -$006c ExNext(lock,fileInfoBlock)(d1/d2)
114 $ff8e -$0072 Info(lock,parameterBlock)(d1/d2)
120 $ff88 -$0078 CreateDir(name)(d1)
126 $ff82 -$007e CurrentDir(lock)(d1)
132 $ff7c -$0084 IoErr()()
138 $ff76 -$008a CreateProc(name,pri,segList,stackSize)(d1/d2/d3/d4)
144 $ff70 -$0090 Exit(returnCode)(d1)
150 $ff6a -$0096 LoadSeg(name)(d1)
156 $ff64 -$009c UnLoadSeg(seglist)(d1)
##private
162 $ff5e -$00a2 ClearVec(bVector,upb)(d1/d2)
168 $ff58 -$00a8 NoReqLoadSeg(bFileName)(d1)
##public
174 $ff52 -$00ae DeviceProc(name)(d1)
180 $ff4c -$00b4 SetComment(name,comment)(d1/d2)
186 $ff46 -$00ba SetProtection(name,protect)(d1/d2)
192 $ff40 -$00c0 DateStamp(date)(d1)
198 $ff3a -$00c6 Delay(timeout)(d1)
204 $ff34 -$00cc WaitForChar(file,timeout)(d1/d2)
210 $ff2e -$00d2 ParentDir(lock)(d1)
216 $ff28 -$00d8 IsInteractive(file)(d1)
222 $ff22 -$00de Execute(string,file,file2)(d1/d2/d3)
*--- functions in V36 or higher (distributed as Release 2.0) ---
*       DOS Object creation/deletion
228 $ff1c -$00e4 AllocDosObject(type,tags)(d1/d2)
```

Function Offset Reference — Page 4

```
234 $ff16 -$00ea FreeDosObject(type,ptr)(d1/d2)
*       Packet Level routines
240 $ff10 -$00f0 DoPkt(port,action,arg1,arg2,arg3,arg4,arg5)
                  (d1/d2/d3/d4/d5/d6/d7)
246 $ff0a -$00f6 SendPkt(dp,port,replyport)(d1/d2/d3)
252 $ff04 -$00fc WaitPkt()()
258 $fefe -$0102 ReplyPkt(dp,res1,res2)(d1/d2/d3)
264 $fef8 -$0108 AbortPkt(port,pkt)(d1/d2)
*       Record Locking
270 $fef2 -$010e LockRecord(fh,offset,length,mode,timeout)(d1/d2/d3/d4/d5)
276 $feec -$0114 LockRecords(recArray,timeout)(d1/d2)
282 $fee6 -$011a UnLockRecord(fh,offset,length)(d1/d2/d3)
288 $fee0 -$0120 UnLockRecords(recArray)(d1)
*       Buffered File I/O
294 $feda -$0126 SelectInput(fh)(d1)
300 $fed4 -$012c SelectOutput(fh)(d1)
306 $fece -$0132 FGetC(fh)(d1)
312 $fec8 -$0138 FPutC(fh,ch)(d1/d2)
318 $fec2 -$013e UnGetC(fh,character)(d1/d2)
324 $febc -$0144 FRead(fh,block,blocklen,number)(d1/d2/d3/d4)
330 $feb6 -$014a FWrite(fh,block,blocklen,number)(d1/d2/d3/d4)
336 $feb0 -$0150 FGets(fh,buf,buflen)(d1/d2/d3)
342 $feaa -$0156 FPuts(fh,str)(d1/d2)
348 $fea4 -$015c VFWritef(fh,format,argarray)(d1/d2/d3)
354 $fe9e -$0162 VFPrintf(fh,format,argarray)(d1/d2/d3)
360 $fe98 -$0168 Flush(fh)(d1)
366 $fe92 -$016e SetVBuf(fh,buff,type,size)(d1/d2/d3/d4)
*       DOS Object Management
372 $fe8c -$0174 DupLockFromFH(fh)(d1)
378 $fe86 -$017a OpenFromLock(lock)(d1)
384 $fe80 -$0180 ParentOfFH(fh)(d1)
390 $fe7a -$0186 ExamineFH(fh,fib)(d1/d2)
396 $fe74 -$018c SetFileDate(name,date)(d1/d2)
402 $fe6e -$0192 NameFromLock(lock,buffer,len)(d1/d2/d3)
408 $fe68 -$0198 NameFromFH(fh,buffer,len)(d1/d2/d3)
414 $fe62 -$019e SplitName(name,seperator,buf,oldpos,size)(d1/d2/d3/d4/d5)
420 $fe5c -$01a4 SameLock(lock1,lock2)(d1/d2)
426 $fe56 -$01aa SetMode(fh,mode)(d1/d2)
432 $fe50 -$01b0 ExAll(lock,buffer,size,data,control)(d1/d2/d3/d4/d5)
438 $fe4a -$01b6 ReadLink(port,lock,path,buffer,size)(d1/d2/d3/d4/d5)
444 $fe44 -$01bc MakeLink(name,dest,soft)(d1/d2/d3)
450 $fe3e -$01c2 ChangeMode(type,fh,newmode)(d1/d2/d3)
456 $fe38 -$01c8 SetFileSize(fh,pos,mode)(d1/d2/d3)
*       Error Handling
462 $fe32 -$01ce SetIoErr(result)(d1)
468 $fe2c -$01d4 Fault(code,header,buffer,len)(d1/d2/d3/d4)
474 $fe26 -$01da PrintFault(code,header)(d1/d2)
480 $fe20 -$01e0 ErrorReport(code,type,arg1,device)(d1/d2/d3/d4)
*--- (1 function slot reserved here) ---
##bias 492
*       Process Management
492 $fe14 -$01ec Cli()()
498 $fe0e -$01f2 CreateNewProc(tags)(d1)
504 $fe08 -$01f8 RunCommand(seg,stack,paramptr,paramlen)(d1/d2/d3/d4)
510 $fe02 -$01fe GetConsoleTask()()
516 $fdfc -$0204 SetConsoleTask(task)(d1)
522 $fdf6 -$020a GetFileSysTask()()
528 $fdf0 -$0210 SetFileSysTask(task)(d1)
534 $fdea -$0216 GetArgStr()()
540 $fde4 -$021c SetArgStr(string)(d1)
546 $fdde -$0222 FindCliProc(num)(d1)
552 $fdd8 -$0228 MaxCli()()
558 $fdd2 -$022e SetCurrentDirName(name)(d1)
564 $fdcc -$0234 GetCurrentDirName(buf,len)(d1/d2)
570 $fdc6 -$023a SetProgramName(name)(d1)
576 $fdc0 -$0240 GetProgramName(buf,len)(d1/d2)
```

Offsets

Function Offset Reference Page 5

```
582 $fdba -$0246 SetPrompt(name)(d1)
588 $fdb4 -$024c GetPrompt(buf,len)(d1/d2)
594 $fdae -$0252 SetProgramDir(lock)(d1)
600 $fda8 -$0258 GetProgramDir()()
*       Device List Management
606 $fda2 -$025e SystemTagList(command,tags)(d1/d2)
612 $fd9c -$0264 AssignLock(name,lock)(d1/d2)
618 $fd96 -$026a AssignLate(name,path)(d1/d2)
624 $fd90 -$0270 AssignPath(name,path)(d1/d2)
630 $fd8a -$0276 AssignAdd(name,lock)(d1/d2)
636 $fd84 -$027c RemAssignList(name,lock)(d1/d2)
642 $fd7e -$0282 GetDeviceProc(name,dp)(d1/d2)
648 $fd78 -$0288 FreeDeviceProc(dp)(d1)
654 $fd72 -$028e LockDosList(flags)(d1)
660 $fd6c -$0294 UnLockDosList(flags)(d1)
666 $fd66 -$029a AttemptLockDosList(flags)(d1)
672 $fd60 -$02a0 RemDosEntry(dlist)(d1)
678 $fd5a -$02a6 AddDosEntry(dlist)(d1)
684 $fd54 -$02ac FindDosEntry(dlist,name,flags)(d1/d2/d3)
690 $fd4e -$02b2 NextDosEntry(dlist,flags)(d1/d2)
696 $fd48 -$02b8 MakeDosEntry(name,type)(d1/d2)
702 $fd42 -$02be FreeDosEntry(dlist)(d1)
708 $fd3c -$02c4 IsFileSystem(name)(d1)
*       Handler Interface
714 $fd36 -$02ca Format(filesystem,volumename,dostype)(d1/d2/d3)
720 $fd30 -$02d0 Relabel(drive,newname)(d1/d2)
726 $fd2a -$02d6 Inhibit(name,onoff)(d1/d2)
732 $fd24 -$02dc AddBuffers(name,number)(d1/d2)
*       Date, Time Routines
738 $fd1e -$02e2 CompareDates(date1,date2)(d1/d2)
744 $fd18 -$02e8 DateToStr(datetime)(d1)
750 $fd12 -$02ee StrToDate(datetime)(d1)
*       Image Management
756 $fd0c -$02f4 InternalLoadSeg(fh,table,funcarray,stack)(d0/a0/a1/a2)
762 $fd06 -$02fa InternalUnLoadSeg(seglist,freefunc)(d1/a1)
768 $fd00 -$0300 NewLoadSeg(file,tags)(d1/d2)
774 $fcfa -$0306 AddSegment(name,seg,system)(d1/d2/d3)
780 $fcf4 -$030c FindSegment(name,seg,system)(d1/d2/d3)
786 $fcee -$0312 RemSegment(seg)(d1)
*       Command Support
792 $fce8 -$0318 CheckSignal(mask)(d1)
798 $fce2 -$031e ReadArgs(template,array,args)(d1/d2/d3)
804 $fcdc -$0324 FindArg(keyword,template)(d1/d2)
810 $fcd6 -$032a ReadItem(name,maxchars,cSource)(d1/d2/d3)
816 $fcd0 -$0330 StrToLong(string,value)(d1/d2)
822 $fcca -$0336 MatchFirst(pat,anchor)(d1/d2)
828 $fcc4 -$033c MatchNext(anchor)(d1)
834 $fcbe -$0342 MatchEnd(anchor)(d1)
840 $fcb8 -$0348 ParsePattern(pat,buf,buflen)(d1/d2/d3)
846 $fcb2 -$034e MatchPattern(pat,str)(d1/d2)
##private
* Not currently implemented.
852 $fcac -$0354 DosNameFromAnchor(anchor,buffer,len)(d1/d2/d3)
##public
858 $fca6 -$035a FreeArgs(args)(d1)
*--- (1 function slot reserved here) ---
##bias 870
870 $fc9a -$0366 FilePart(path)(d1)
876 $fc94 -$036c PathPart(path)(d1)
882 $fc8e -$0372 AddPart(dirname,filename,size)(d1/d2/d3)
*       Notification
888 $fc88 -$0378 StartNotify(notify)(d1)
894 $fc82 -$037e EndNotify(notify)(d1)
*       Environment Variable functions
900 $fc7c -$0384 SetVar(name,buffer,size,flags)(d1/d2/d3/d4)
906 $fc76 -$038a GetVar(name,buffer,size,flags)(d1/d2/d3/d4)
```

Function Offset Reference Page 6

```
912 $fc70 -$0390 DeleteVar(name,flags)(d1/d2)
918 $fc6a -$0396 FindVar(name,type)(d1/d2)
924 $fc64 -$039c CliInit(dp)(a0)
930 $fc5e -$03a2 CliInitNewcli(dp)(a0)
936 $fc58 -$03a8 CliInitRun(dp)(a0)
942 $fc52 -$03ae WriteChars(buf,buflen)(d1/d2)
948 $fc4c -$03b4 PutStr(str)(d1)
954 $fc46 -$03ba VPrintf(format,argarray)(d1/d2)
*--- (1 function slot reserved here) ---
##bias 966
* these were unimplemented until dos 36.147
966 $fc3a -$03c6 ParsePatternNoCase(pat,buf,buflen)(d1/d2/d3)
972 $fc34 -$03cc MatchPatternNoCase(pat,str)(d1/d2)
##private
978 $fc2e -$03d2 DosGetString(num)(d1)
##public
* this was added for V37 dos, returned 0 before then.
984 $fc28 -$03d8 SameDevice(lock1,lock2)(d1/d2)
* These were added in dos 36.147
*--- (4 function slots reserved here) ---
##bias 1014
* these were added in dos 37.1
*--- (2 function slots reserved here) ---
##bias 1026
* these were added in dos 37.8
*--- (2 function slots reserved here) ---
##bias 1038
##end

***************** exec
##base _SysBase
##bias 30
##public
*------ misc -----------------------------------------------------
 30 $ffe2 -$001e Supervisor(userFunction)(a5)
*------ special patchable hooks to internal exec activity ------------
##private
 36 $ffdc -$0024 ExitIntr()()
 42 $ffd6 -$002a Schedule()()
 48 $ffd0 -$0030 Reschedule()()
 54 $ffca -$0036 Switch()()
 60 $ffc4 -$003c Dispatch()()
 66 $ffbe -$0042 Exception()()
##public
*------ module creation -------------------------------------------
 72 $ffb8 -$0048 InitCode(startClass,version)(d0/d1)
 78 $ffb2 -$004e InitStruct(initTable,memory,size)(a1/a2,d0)
 84 $ffac -$0054 MakeLibrary(funcInit,structInit,libInit,dataSize,segList)
                 (a0/a1/a2,d0/d1)
 90 $ffa6 -$005a MakeFunctions(target,functionArray,funcDispBase)(a0/a1/a2)
 96 $ffa0 -$0060 FindResident(name)(a1)
102 $ff9a -$0066 InitResident(resident,segList)(a1,d1)
*------ diagnostics -----------------------------------------------
108 $ff94 -$006c Alert(alertNum)(d7)
114 $ff8e -$0072 Debug(flags)(d0)
*------ interrupts ------------------------------------------------
120 $ff88 -$0078 Disable()()
126 $ff82 -$007e Enable()()
132 $ff7c -$0084 Forbid()()
138 $ff76 -$008a Permit()()
144 $ff70 -$0090 SetSR(newSR,mask)(d0/d1)
150 $ff6a -$0096 SuperState()()
156 $ff64 -$009c UserState(sysStack)(d0)
162 $ff5e -$00a2 SetIntVector(intNumber,interrupt)(d0/a1)
168 $ff58 -$00a8 AddIntServer(intNumber,interrupt)(d0/a1)
174 $ff52 -$00ae RemIntServer(intNumber,interrupt)(d0/a1)
```

Function Offset Reference Page 7

```
180 $ff4c -$00b4 Cause(interrupt)(a1)
*------ memory allocation ------------------------------------------
186 $ff46 -$00ba Allocate(freeList,byteSize)(a0,d0)
192 $ff40 -$00c0 Deallocate(freeList,memoryBlock,byteSize)(a0/a1,d0)
198 $ff3a -$00c6 AllocMem(byteSize,requirements)(d0/d1)
204 $ff34 -$00cc AllocAbs(byteSize,location)(d0/a1)
210 $ff2e -$00d2 FreeMem(memoryBlock,byteSize)(a1,d0)
216 $ff28 -$00d8 AvailMem(requirements)(d1)
222 $ff22 -$00de AllocEntry(entry)(a0)
228 $ff1c -$00e4 FreeEntry(entry)(a0)
*------ lists ------------------------------------------------------
234 $ff16 -$00ea Insert(list,node,pred)(a0/a1/a2)
240 $ff10 -$00f0 AddHead(list,node)(a0/a1)
246 $ff0a -$00f6 AddTail(list,node)(a0/a1)
252 $ff04 -$00fc Remove(node)(a1)
258 $fefe -$0102 RemHead(list)(a0)
264 $fef8 -$0108 RemTail(list)(a0)
270 $fef2 -$010e Enqueue(list,node)(a0/a1)
276 $feec -$0114 FindName(list,name)(a0/a1)
*------ tasks ------------------------------------------------------
282 $fee6 -$011a AddTask(task,initPC,finalPC)(a1/a2/a3)
288 $fee0 -$0120 RemTask(task)(a1)
294 $feda -$0126 FindTask(name)(a1)
300 $fed4 -$012c SetTaskPri(task,priority)(a1,d0)
306 $fece -$0132 SetSignal(newSignals,signalSet)(d0/d1)
312 $fec8 -$0138 SetExcept(newSignals,signalSet)(d0/d1)
318 $fec2 -$013e Wait(signalSet)(d0)
324 $febc -$0144 Signal(task,signalSet)(a1,d0)
330 $feb6 -$014a AllocSignal(signalNum)(d0)
336 $feb0 -$0150 FreeSignal(signalNum)(d0)
342 $feaa -$0156 AllocTrap(trapNum)(d0)
348 $fea4 -$015c FreeTrap(trapNum)(d0)
*------ messages ---------------------------------------------------
354 $fe9e -$0162 AddPort(port)(a1)
360 $fe98 -$0168 RemPort(port)(a1)
366 $fe92 -$016e PutMsg(port,message)(a0/a1)
372 $fe8c -$0174 GetMsg(port)(a0)
378 $fe86 -$017a ReplyMsg(message)(a1)
384 $fe80 -$0180 WaitPort(port)(a0)
390 $fe7a -$0186 FindPort(name)(a1)
*------ libraries --------------------------------------------------
396 $fe74 -$018c AddLibrary(library)(a1)
402 $fe6e -$0192 RemLibrary(library)(a1)
408 $fe68 -$0198 OldOpenLibrary(libName)(a1)
414 $fe62 -$019e CloseLibrary(library)(a1)
420 $fe5c -$01a4 SetFunction(library,funcOffset,newFunction)(a1,a0,d0)
426 $fe56 -$01aa SumLibrary(library)(a1)
*------ devices ----------------------------------------------------
432 $fe50 -$01b0 AddDevice(device)(a1)
438 $fe4a -$01b6 RemDevice(device)(a1)
444 $fe44 -$01bc OpenDevice(devName,unit,ioRequest,flags)(a0,d0/a1,d1)
450 $fe3e -$01c2 CloseDevice(ioRequest)(a1)
456 $fe38 -$01c8 DoIO(ioRequest)(a1)
462 $fe32 -$01ce SendIO(ioRequest)(a1)
468 $fe2c -$01d4 CheckIO(ioRequest)(a1)
474 $fe26 -$01da WaitIO(ioRequest)(a1)
480 $fe20 -$01e0 AbortIO(ioRequest)(a1)
*------ resources --------------------------------------------------
486 $fe1a -$01e6 AddResource(resource)(a1)
492 $fe14 -$01ec RemResource(resource)(a1)
498 $fe0e -$01f2 OpenResource(resName)(a1)
*------ private diagnostic support ---------------------------------
##private
504 $fe08 -$01f8 RawIOInit()()
510 $fe02 -$01fe RawMayGetChar()()
516 $fdfc -$0204 RawPutChar()()
```

Function Offset Reference Page 8

```
##public
*------ misc -------------------------------------------------------
522 $fdf6 -$020a RawDoFmt(formatString,dataStream,putChProc,putChData)
                 (a0/a1/a2/a3)
528 $fdf0 -$0210 GetCC()()
534 $fdea -$0216 TypeOfMem(address)(a1)
540 $fde4 -$021c Procure(semaport,bidMsg)(a0/a1)
546 $fdde -$0222 Vacate(semaport)(a0)
552 $fdd8 -$0228 OpenLibrary(libName,version)(a1,d0)
*--- functions in V33 or higher (distributed as Release 1.2) ---
*------ signal semaphores (note funny registers)--------------------
558 $fdd2 -$022e InitSemaphore(sigSem)(a0)
564 $fdcc -$0234 ObtainSemaphore(sigSem)(a0)
570 $fdc6 -$023a ReleaseSemaphore(sigSem)(a0)
576 $fdc0 -$0240 AttemptSemaphore(sigSem)(a0)
582 $fdba -$0246 ObtainSemaphoreList(sigSem)(a0)
588 $fdb4 -$024c ReleaseSemaphoreList(sigSem)(a0)
594 $fdae -$0252 FindSemaphore(sigSem)(a1)
600 $fda8 -$0258 AddSemaphore(sigSem)(a1)
606 $fda2 -$025e RemSemaphore(sigSem)(a1)
*------ kickmem support --------------------------------------------
612 $fd9c -$0264 SumKickData()()
*------ more memory support ----------------------------------------
618 $fd96 -$026a AddMemList(size,attributes,pri,base,name)(d0/d1/d2/a0/a1)
624 $fd90 -$0270 CopyMem(source,dest,size)(a0/a1,d0)
630 $fd8a -$0276 CopyMemQuick(source,dest,size)(a0/a1,d0)
*------ cache ------------------------------------------------------
*--- functions in V36 or higher (distributed as Release 2.0) ---
636 $fd84 -$027c CacheClearU()()
642 $fd7e -$0282 CacheClearE(address,length,caches)(a0,d0/d1)
648 $fd78 -$0288 CacheControl(cacheBits,cacheMask)(d0/d1)
*------ misc -------------------------------------------------------
654 $fd72 -$028e CreateIORequest(port,size)(a0,d0)
660 $fd6c -$0294 DeleteIORequest(iorequest)(a0)
666 $fd66 -$029a CreateMsgPort()()
672 $fd60 -$02a0 DeleteMsgPort(port)(a0)
678 $fd5a -$02a6 ObtainSemaphoreShared(sigSem)(a0)
*------ even more memory support -----------------------------------
684 $fd54 -$02ac AllocVec(byteSize,requirements)(d0/d1)
690 $fd4e -$02b2 FreeVec(memoryBlock)(a1)
696 $fd48 -$02b8 CreatePrivatePool(requirements,puddleSize,puddleThresh)
                 (d0/d1/d2)
702 $fd42 -$02be DeletePrivatePool(poolHeader)(a0)
708 $fd3c -$02c4 AllocPooled(memSize,poolHeader)(d0/a0)
714 $fd36 -$02ca FreePooled(memory,poolHeader)(a1,a0)
*------ misc -------------------------------------------------------
##private
720 $fd30 -$02d0 ExecReserved00(nothing)(d0)
##public
726 $fd2a -$02d6 ColdReboot()()
732 $fd24 -$02dc StackSwap(newSize,newSP,newStack)(d0/d1/a0)
*------ task trees -------------------------------------------------
738 $fd1e -$02e2 ChildFree(tid)(d0)
744 $fd18 -$02e8 ChildOrphan(tid)(d0)
750 $fd12 -$02ee ChildStatus(tid)(d0)
756 $fd0c -$02f4 ChildWait(tid)(d0)
*------ future expansion -------------------------------------------
##private
762 $fd06 -$02fa ExecReserved01(nothing)(d0)
768 $fd00 -$0300 ExecReserved02(nothing)(d0)
774 $fcfa -$0306 ExecReserved03(nothing)(d0)
780 $fcf4 -$030c ExecReserved04(nothing)(d0)
##public
##end

***************** expansion
```

Function Offset Reference

```
##base _ExpansionBase
##bias 30
##public
*--- functions in V33 or higher (distributed as Release 1.2) ---
 30 $ffe2 -$001e AddConfigDev(configDev)(a0)
*--- functions in V36 or higher (distributed as Release 2.0) ---
 36 $ffdc -$0024 AddBootNode(bootPri,flags,deviceNode,configDev)
                (d0/d1/a0/a1)
*--- functions in V33 or higher (distributed as Release 1.2) ---
 42 $ffd6 -$002a AllocBoardMem(slotSpec)(d0)
 48 $ffd0 -$0030 AllocConfigDev()()
 54 $ffca -$0036 AllocExpansionMem(numSlots,slotAlign)(d0/d1)
 60 $ffc4 -$003c ConfigBoard(board,configDev)(a0/a1)
 66 $ffbe -$0042 ConfigChain(baseAddr)(a0)
 72 $ffb8 -$0048 FindConfigDev(oldConfigDev,manufacturer,product)(a0,d0/d1)
 78 $ffb2 -$004e FreeBoardMem(startSlot,slotSpec)(d0/d1)
 84 $ffac -$0054 FreeConfigDev(configDev)(a0)
 90 $ffa6 -$005a FreeExpansionMem(startSlot,numSlots)(d0/d1)
 96 $ffa0 -$0060 ReadExpansionByte(board,offset)(a0,d0)
102 $ff9a -$0066 ReadExpansionRom(board,configDev)(a0/a1)
108 $ff94 -$006c RemConfigDev(configDev)(a0)
114 $ff8e -$0072 WriteExpansionByte(board,offset,byte)(a0,d0/d1)
120 $ff88 -$0078 ObtainConfigBinding()()
126 $ff82 -$007e ReleaseConfigBinding()()
132 $ff7c -$0084 SetCurrentBinding(currentBinding,bindingSize)(a0,d0)
138 $ff76 -$008a GetCurrentBinding(currentBinding,bindingSize)(a0,d0)
144 $ff70 -$0090 MakeDosNode(parmPacket)(a0)
150 $ff6a -$0096 AddDosNode(bootPri,flags,deviceNode)(d0/d1/a0)
##private
*--- functions in V36 or higher (distributed as Release 2.0) ---
156 $ff64 -$009c ExpansionReserved26()()
162 $ff5e -$00a2 WriteExpansionWord(board,offset,word)(a0,d0/d1)
##end

***************** gadtools
* "gadtools.library"
##base _GadToolsBase
##bias 30
##public
*--- functions in V36 or higher (distributed as Release 2.0) ---
*
* Gadget Functions
*
 30 $ffe2 -$001e CreateGadgetA(kind,gad,ng,taglist)(d0/a0/a1/a2)
 36 $ffdc -$0024 FreeGadgets(gad)(a0)
 42 $ffd6 -$002a GT_SetGadgetAttrsA(gad,win,req,taglist)(a0/a1/a2/a3)
*
* Menu functions
*
 48 $ffd0 -$0030 CreateMenusA(newmenu,taglist)(a0/a1)
 54 $ffca -$0036 FreeMenus(menu)(a0)
 60 $ffc4 -$003c LayoutMenuItemsA(firstitem,vi,taglist)(a0/a1/a2)
 66 $ffbe -$0042 LayoutMenusA(firstmenu,vi,taglist)(a0/a1/a2)
*
* Misc Event-Handling Functions
*
 72 $ffb8 -$0048 GT_GetIMsg(iport)(a0)
 78 $ffb2 -$004e GT_ReplyIMsg(imsg)(a1)
 84 $ffac -$0054 GT_RefreshWindow(win,req)(a0/a1)
 90 $ffa6 -$005a GT_BeginRefresh(win)(a0)
 96 $ffa0 -$0060 GT_EndRefresh(win,complete)(a0,d0)
102 $ff9a -$0066 GT_FilterIMsg(imsg)(a1)
108 $ff94 -$006c GT_PostFilterIMsg(imsg)(a1)
114 $ff8e -$0072 CreateContext(glistptr)(a0)
*
* Rendering Functions
```

Function Offset Reference

```
*
120 $ff88 -$0078 DrawBevelBoxA(rport,left,top,width,height,taglist)
                 (a0,d0/d1/d2/d3/a1)
*
* Visuals Functions
*
126 $ff82 -$007e GetVisualInfoA(screen,taglist)(a0/a1)
132 $ff7c -$0084 FreeVisualInfo(vi)(a0)
##private
*
* Reserved entries
*
138 $ff76 -$008a GTReserved0()()
144 $ff70 -$0090 GTReserved1()()
150 $ff6a -$0096 GTReserved2()()
156 $ff64 -$009c GTReserved3()()
162 $ff5e -$00a2 GTReserved4()()
168 $ff58 -$00a8 GTReserved5()()
##end

***************** graphics
* "graphics.library"
##base _GfxBase
##bias 30
##public
*------ BitMap primitives ------
 30 $ffe2 -$001e BltBitMap(srcBitMap,xSrc,ySrc,destBitMap,xDest,yDest,xSize,
                 ySize,minterm,mask,tempA)(a0,d0/d1/a1,d2/d3/d4/d5/d6/d7/a2)
 36 $ffdc -$0024 BltTemplate(source,xSrc,srcMod,destRP,xDest,yDest,xSize,
                 ySize)(a0,d0/d1/a1,d2/d3/d4/d5)
*------ Text routines ------
 42 $ffd6 -$002a ClearEOL(rp)(a1)
 48 $ffd0 -$0030 ClearScreen(rp)(a1)
 54 $ffca -$0036 TextLength(rp,string,count)(a1,a0,d0)
 60 $ffc4 -$003c Text(rp,string,count)(a1,a0,d0)
 66 $ffbe -$0042 SetFont(rp,textFont)(a1,a0)
 72 $ffb8 -$0048 OpenFont(textAttr)(a0)
 78 $ffb2 -$004e CloseFont(textFont)(a1)
 84 $ffac -$0054 AskSoftStyle(rp)(a1)
 90 $ffa6 -$005a SetSoftStyle(rp,style,enable)(a1,d0/d1)
*------ Gels routines ------
 96 $ffa0 -$0060 AddBob(bob,rp)(a0/a1)
102 $ff9a -$0066 AddVSprite(vSprite,rp)(a0/a1)
108 $ff94 -$006c DoCollision(rp)(a1)
114 $ff8e -$0072 DrawGList(rp,vp)(a1,a0)
120 $ff88 -$0078 InitGels(head,tail,gelsInfo)(a0/a1/a2)
126 $ff82 -$007e InitMasks(vSprite)(a0)
132 $ff7c -$0084 RemIBob(bob,rp,vp)(a0/a1/a2)
138 $ff76 -$008a RemVSprite(vSprite)(a0)
144 $ff70 -$0090 SetCollision(num,routine,gelsInfo)(d0/a0/a1)
150 $ff6a -$0096 SortGList(rp)(a1)
156 $ff64 -$009c AddAnimOb(anOb,anKey,rp)(a0/a1/a2)
162 $ff5e -$00a2 Animate(anKey,rp)(a0/a1)
168 $ff58 -$00a8 GetGBuffers(anOb,rp,flag)(a0/a1,d0)
174 $ff52 -$00ae InitGMasks(anOb)(a0)
*------ General graphics routines ------
180 $ff4c -$00b4 DrawEllipse(rp,xCenter,yCenter,a,b)(a1,d0/d1/d2/d3)
186 $ff46 -$00ba AreaEllipse(rp,xCenter,yCenter,a,b)(a1,d0/d1/d2/d3)
192 $ff40 -$00c0 LoadRGB4(vp,colors,count)(a0/a1,d0)
198 $ff3a -$00c6 InitRastPort(rp)(a1)
204 $ff34 -$00cc InitVPort(vp)(a0)
210 $ff2e -$00d2 MrgCop(view)(a1)
216 $ff28 -$00d8 MakeVPort(view,vp)(a0/a1)
222 $ff22 -$00de LoadView(view)(a1)
228 $ff1c -$00e4 WaitBlit()()
234 $ff16 -$00ea SetRast(rp,pen)(a1,d0)
```

Function Offset Reference

Page 11

```
240 $ff10 -$00f0 Move(rp,x,y)(a1,d0/d1)
246 $ff0a -$00f6 Draw(rp,x,y)(a1,d0/d1)
252 $ff04 -$00fc AreaMove(rp,x,y)(a1,d0/d1)
258 $fefe -$0102 AreaDraw(rp,x,y)(a1,d0/d1)
264 $fef8 -$0108 AreaEnd(rp)(a1)
270 $fef2 -$010e WaitTOF()()
276 $feec -$0114 QBlit(blit)(a1)
282 $fee6 -$011a InitArea(areaInfo,vectorBuffer,maxVectors)(a0/a1,d0)
288 $fee0 -$0120 SetRGB4(vp,index,red,green,blue)(a0,d0/d1/d2/d3)
294 $feda -$0126 QBSBlit(blit)(a1)
300 $fed4 -$012c BltClear(memBlock,byteCount,flags)(a1,d0/d1)
306 $fece -$0132 RectFill(rp,xMin,yMin,xMax,yMax)(a1,d0/d1/d2/d3)
312 $fec8 -$0138 BltPattern(rp,mask,xMin,yMin,xMax,yMax,maskBPR)
                 (a1,a0,d0/d1/d2/d3/d4)
318 $fec2 -$013e ReadPixel(rp,x,y)(a1,d0/d1)
324 $febc -$0144 WritePixel(rp,x,y)(a1,d0/d1)
330 $feb6 -$014a Flood(rp,mode,x,y)(a1,d2,d0/d1)
336 $feb0 -$0150 PolyDraw(rp,count,polyTable)(a1,d0/a0)
342 $feaa -$0156 SetAPen(rp,pen)(a1,d0)
348 $fea4 -$015c SetBPen(rp,pen)(a1,d0)
354 $fe9e -$0162 SetDrMd(rp,drawMode)(a1,d0)
360 $fe98 -$0168 InitView(view)(a1)
366 $fe92 -$016e CBump(copList)(a1)
372 $fe8c -$0174 CMove(copList,destination,data)(a1,d0/d1)
378 $fe86 -$017a CWait(copList,v,h)(a1,d0/d1)
384 $fe80 -$0180 VBeamPos()()
390 $fe7a -$0186 InitBitMap(bitMap,depth,width,height)(a0,d0/d1/d2)
396 $fe74 -$018c ScrollRaster(rp,dx,dy,xMin,yMin,xMax,yMax)
                 (a1,d0/d1/d2/d3/d4/d5)
402 $fe6e -$0192 WaitBOVP(vp)(a0)
408 $fe68 -$0198 GetSprite(sprite,num)(a0,d0)
414 $fe62 -$019e FreeSprite(num)(d0)
420 $fe5c -$01a4 ChangeSprite(vp,sprite,newData)(a0/a1/a2)
426 $fe56 -$01aa MoveSprite(vp,sprite,x,y)(a0/a1,d0/d1)
432 $fe50 -$01b0 LockLayerRom(layer)(a5)
438 $fe4a -$01b6 UnlockLayerRom(layer)(a5)
444 $fe44 -$01bc SyncSBitMap(layer)(a0)
450 $fe3e -$01c2 CopySBitMap(layer)(a0)
456 $fe38 -$01c8 OwnBlitter()()
462 $fe32 -$01ce DisownBlitter()()
468 $fe2c -$01d4 InitTmpRas(tmpRas,buffer,size)(a0/a1,d0)
474 $fe26 -$01da AskFont(rp,textAttr)(a1,a0)
480 $fe20 -$01e0 AddFont(textFont)(a1)
486 $fe1a -$01e6 RemFont(textFont)(a1)
492 $fe14 -$01ec AllocRaster(width,height)(d0/d1)
498 $fe0e -$01f2 FreeRaster(p,width,height)(a0,d0/d1)
504 $fe08 -$01f8 AndRectRegion(region,rectangle)(a0/a1)
510 $fe02 -$01fe OrRectRegion(region,rectangle)(a0/a1)
516 $fdfc -$0204 NewRegion()()
522 $fdf6 -$020a ClearRectRegion(region,rectangle)(a0/a1)
528 $fdf0 -$0210 ClearRegion(region)(a0)
534 $fdea -$0216 DisposeRegion(region)(a0)
540 $fde4 -$021c FreeVPortCopLists(vp)(a0)
546 $fdde -$0222 FreeCopList(copList)(a0)
552 $fdd8 -$0228 ClipBlit(srcRP,xSrc,ySrc,destRP,xDest,yDest,xSize,ySize,
                  minterm)(a0,d0/d1/a1,d2/d3/d4/d5/d6)
558 $fdd2 -$022e XorRectRegion(region,rectangle)(a0/a1)
564 $fdcc -$0234 FreeCprList(cprList)(a0)
570 $fdc6 -$023a GetColorMap(entries)(d0)
576 $fdc0 -$0240 FreeColorMap(colorMap)(a0)
582 $fdba -$0246 GetRGB4(colorMap,entry)(a0,d0)
588 $fdb4 -$024c ScrollVPort(vp)(a0)
594 $fdae -$0252 UCopperListInit(uCopList,n)(a0,d0)
600 $fda8 -$0258 FreeGBuffers(anOb,rp,flag)(a0/a1,d0)
606 $fda2 -$025e BltBitMapRastPort(srcBitMap,xSrc,ySrc,destRP,xDest,yDest,
                  xSize,ySize,minterm)(a0,d0/d1/a1,d2/d3/d4/d5/d6)
```

Function Offset Reference

Page 12

```
612 $fd9c -$0264 OrRegionRegion(srcRegion,destRegion)(a0/a1)
618 $fd96 -$026a XorRegionRegion(srcRegion,destRegion)(a0/a1)
624 $fd90 -$0270 AndRegionRegion(srcRegion,destRegion)(a0/a1)
630 $fd8a -$0276 SetRGB4CM(colorMap,index,red,green,blue)(a0,d0/d1/d2/d3)
636 $fd84 -$027c BltMaskBitMapRastPort(srcBitMap,xSrc,ySrc,destRP,xDest,
                  yDest,xSize,ySize,minterm,bltMask)(a0,d0/d1/a1,d2/d3/d4/d5/d6/a2)
*--- (2 function slots reserved here) ---
##bias 654
654 $fd72 -$028e AttemptLockLayerRom(layer)(a5)
*--- functions in V36 or higher (distributed as Release 2.0) ---
660 $fd6c -$0294 GfxNew(gfxNodeType)(d0)
666 $fd66 -$029a GfxFree(gfxNodePtr)(a0)
672 $fd60 -$02a0 GfxAssociate(associateNode,gfxNodePtr)(a0/a1)
678 $fd5a -$02a6 BitMapScale(bitScaleArgs)(a0)
684 $fd54 -$02ac ScalerDiv(factor,numerator,denominator)(d0/d1/d2)
690 $fd4e -$02b2 TextExtent(rp,string,count,textExtent)(a1,a0,d0/a2)
696 $fd48 -$02b8 TextFit(rp,string,strLen,textExtent,constrainingExtent,
                  strDirection,constrainingBitWidth,constrainingBitHeight)(a1,a0,d0
/a2/a3,d1/d2/d3)
702 $fd42 -$02be GfxLookUp(associateNode)(a0)
708 $fd3c -$02c4 VideoControl(colorMap,tagarray)(a0/a1)
714 $fd36 -$02ca OpenMonitor(monitorName,displayID)(a1,d0)
720 $fd30 -$02d0 CloseMonitor(monitorSpec)(a0)
726 $fd2a -$02d6 FindDisplayInfo(displayID)(d0)
732 $fd24 -$02dc NextDisplayInfo(displayID)(d0)
##private
738 $fd1e -$02e2 AddDisplayInfo(displayInfoRecord)(a0)
744 $fd18 -$02e8 AddDisplayInfoData(handle,buf,size,tagID,displayID)
                  (a0/a1,d0/d1/d2)
750 $fd12 -$02ee SetDisplayInfoData(handle,buf,size,tagID,displayID)
                  (a0/a1,d0/d1/d2)
##public
756 $fd0c -$02f4 GetDisplayInfoData(handle,buf,size,tagID,displayID)
                  (a0/a1,d0/d1/d2)
762 $fd06 -$02fa FontExtent(font,fontExtent)(a0/a1)
768 $fd00 -$0300 ReadPixelLine8(rp,xstart,ystart,width,array,tempRP)
                  (a0,d0/d1/d2/a2,a1)
774 $fcfa -$0306 WritePixelLine8(rp,xstart,ystart,width,array,tempRP)
                  (a0,d0/d1/d2/a2,a1)
780 $fcf4 -$030c ReadPixelArray8(rp,xstart,ystart,xstop,ystop,array,temprp)
                  (a0,d0/d1/d2/d3/a2,a1)
786 $fcee -$0312 WritePixelArray8(rp,xstart,ystart,xstop,ystop,array,temprp)
                  (a0,d0/d1/d2/d3/a2,a1)
792 $fce8 -$0318 GetVPModeID(vp)(a0)
798 $fce2 -$031e ModeNotAvailable(modeID)(d0)
804 $fcdc -$0324 WeighTAMatch(reqTextAttr,targetTextAttr,targetTags)
                  (a0/a1/a2)
810 $fcd6 -$032a EraseRect(rp,xMin,yMin,xMax,yMax)(a1,d0/d1/d2/d3)
816 $fcd0 -$0330 ExtendFont(font,fontTags)(a0/a1)
822 $fcca -$0336 StripFont(font)(a0)
##end

***************** icon
* "icon.library"
##base _IconBase
##bias 30
*--- functions in V36 or higher (distributed as Release 2.0) ---
*       Use DiskObjects instead of obsolete WBObjects
##private
 30 $ffe2 -$001e OBSOLETEGetWBObject(name)(a0)
 36 $ffdc -$0024 OBSOLETEPutWBObject(name,object)(a0/a1)
##public
 42 $ffd6 -$002a GetIcon(name,icon,freelist)(a0/a1/a2)
 48 $ffd0 -$0030 PutIcon(name,icon)(a0/a1)
 54 $ffca -$0036 FreeFreeList(freelist)(a0)
##private
```

Function Offset Reference Page 13

```
 60 $ffc4 -$003c OBSOLETEFreeWBObject(object)(a0)
 66 $ffbe -$0042 OBSOLETEAllocWBObject()()
##public
 72 $ffb8 -$0048 AddFreeList(freelist,mem,size)(a0/a1/a2)
 78 $ffb2 -$004e GetDiskObject(name)(a0)
 84 $ffac -$0054 PutDiskObject(name,diskobj)(a0/a1)
 90 $ffa6 -$005a FreeDiskObject(diskobj)(a0)
 96 $ffa0 -$0060 FindToolType(toolTypeArray,typeName)(a0/a1)
102 $ff9a -$0066 MatchToolValue(typeString,value)(a0/a1)
108 $ff94 -$006c BumpRevision(newname,oldname)(a0/a1)
##private
114 $ff8e -$0072 FreeAlloc(free,len,type)(a0/a1/a2)
##public
120 $ff88 -$0078 GetDefDiskObject(type)(d0)
126 $ff82 -$007e PutDefDiskObject(diskObject)(a0)
132 $ff7c -$0084 GetDiskObjectNew(name)(a0)
138 $ff76 -$008a DeleteDiskObject(name)(a0)
*--- (4 function slots reserved here) ---
##bias 168
##end

***************** iffparse
* "iffparse.library"
##base _IFFParseBase
##bias 30
##public
*--- functions in V33 or higher (distributed as Release 1.2) ---
*------ Basic functions ------
 30 $ffe2 -$001e AllocIFF()()
 36 $ffdc -$0024 OpenIFF(iff,rwMode)(a0,d0)
 42 $ffd6 -$002a ParseIFF(iff,control)(a0,d0)
 48 $ffd0 -$0030 CloseIFF(iff)(a0)
 54 $ffca -$0036 FreeIFF(iff)(a0)
*------ Read/Write functions ------
 60 $ffc4 -$003c ReadChunkBytes(iff,buf,size)(a0/a1,d0)
 66 $ffbe -$0042 WriteChunkBytes(iff,buf,size)(a0/a1,d0)
 72 $ffb8 -$0048 ReadChunkRecords(iff,buf,bytesPerRecord,nRecords)
                 (a0/a1,d0/d1)
 78 $ffb2 -$004e WriteChunkRecords(iff,buf,bytesPerRecord,nRecords)
                 (a0/a1,d0/d1)
*------ Context entry/exit ------
 84 $ffac -$0054 PushChunk(iff,type,id,size)(a0,d0/d1/d2)
 90 $ffa6 -$005a PopChunk(iff)(a0)
*--- (1 function slot reserved here) ---
##bias 102
*------ Low-level handler installation ------
102 $ff9a -$0066 EntryHandler(iff,type,id,position,handler,object)
                 (a0,d0/d1/d2/a1/a2)
108 $ff94 -$006c ExitHandler(iff,type,id,position,handler,object)
                 (a0,d0/d1/d2/a1/a2)
*------ Built-in chunk/property handlers ------
114 $ff8e -$0072 PropChunk(iff,type,id)(a0,d0/d1)
120 $ff88 -$0078 PropChunks(iff,propArray,nProps)(a0/a1,d0)
126 $ff82 -$007e StopChunk(iff,type,id)(a0,d0/d1)
132 $ff7c -$0084 StopChunks(iff,propArray,nProps)(a0/a1,d0)
138 $ff76 -$008a CollectionChunk(iff,type,id)(a0,d0/d1)
144 $ff70 -$0090 CollectionChunks(iff,propArray,nProps)(a0/a1,d0)
150 $ff6a -$0096 StopOnExit(iff,type,id)(a0,d0/d1)
*------ Context utilities ------
156 $ff64 -$009c FindProp(iff,type,id)(a0,d0/d1)
162 $ff5e -$00a2 FindCollection(iff,type,id)(a0,d0/d1)
168 $ff58 -$00a8 FindPropContext(iff)(a0)
174 $ff52 -$00ae CurrentChunk(iff)(a0)
180 $ff4c -$00b4 ParentChunk(contextNode)(a0)
*------ LocalContextItem support functions ------
186 $ff46 -$00ba AllocLocalItem(type,id,ident,dataSize)(d0/d1/d2/d3)
```

Function Offset Reference Page 14

```
192 $ff40 -$00c0 LocalItemData(localItem)(a0)
198 $ff3a -$00c6 SetLocalItemPurge(localItem,purgeHook)(a0/a1)
204 $ff34 -$00cc FreeLocalItem(localItem)(a0)
210 $ff2e -$00d2 FindLocalItem(iff,type,id,ident)(a0,d0/d1/d2)
216 $ff28 -$00d8 StoreLocalItem(iff,localItem,position)(a0/a1,d0)
222 $ff22 -$00de StoreItemInContext(iff,localItem,contextNode)(a0/a1/a2)
*------ IFFHandle initialization ------
228 $ff1c -$00e4 InitIFF(iff,flags,streamHook)(a0,d0/a1)
234 $ff16 -$00ea InitIFFasDOS(iff)(a0)
240 $ff10 -$00f0 InitIFFasClip(iff)(a0)
*------ Internal clipboard support ------
246 $ff0a -$00f6 OpenClipboard(unitNum)(d0)
252 $ff04 -$00fc CloseClipboard(clipboard)(a0)
*------ Miscellaneous ------
258 $fefe -$0102 GoodID(id)(d0)
264 $fef8 -$0108 GoodType(type)(d0)
270 $fef2 -$010e IDtoStr(id,buf)(d0/a0)
##end

***************** input
* "input.device"
##base _InputBase
##bias 42
##public
*--- functions in V36 or higher (distributed as Release 2.0) ---
 42 $ffd6 -$002a PeekQualifier()()
##end

***************** intuition
* "intuition.library"
##base _IntuitionBase
##bias 30
##public
* Public functions OpenIntuition() and Intuition() are intentionally
* not documented.
 30 $ffe2 -$001e OpenIntuition()()
 36 $ffdc -$0024 Intuition(iEvent)(a0)
 42 $ffd6 -$002a AddGadget(window,gadget,position)(a0/a1,d0)
 48 $ffd0 -$0030 ClearDMRequest(window)(a0)
 54 $ffca -$0036 ClearMenuStrip(window)(a0)
 60 $ffc4 -$003c ClearPointer(window)(a0)
 66 $ffbe -$0042 CloseScreen(screen)(a0)
 72 $ffb8 -$0048 CloseWindow(window)(a0)
 78 $ffb2 -$004e CloseWorkBench()()
 84 $ffac -$0054 CurrentTime(seconds,micros)(a0/a1)
 90 $ffa6 -$005a DisplayAlert(alertNumber,string,height)(d0/a0,d1)
 96 $ffa0 -$0060 DisplayBeep(screen)(a0)
102 $ff9a -$0066 DoubleClick(sSeconds,sMicros,cSeconds,cMicros)(d0/d1/d2/d3)
108 $ff94 -$006c DrawBorder(rp,border,leftOffset,topOffset)(a0/a1,d0/d1)
114 $ff8e -$0072 DrawImage(rp,image,leftOffset,topOffset)(a0/a1,d0/d1)
120 $ff88 -$0078 EndRequest(requester,window)(a0/a1)
126 $ff82 -$007e GetDefPrefs(preferences,size)(a0,d0)
132 $ff7c -$0084 GetPrefs(preferences,size)(a0,d0)
138 $ff76 -$008a InitRequester(requester)(a0)
144 $ff70 -$0090 ItemAddress(menuStrip,menuNumber)(a0,d0)
150 $ff6a -$0096 ModifyIDCMP(window,flags)(a0,d0)
156 $ff64 -$009c ModifyProp(gadget,window,requester,flags,horizPot,vertPot,
                 horizBody,vertBody)(a0/a1/a2,d0/d1/d2/d3/d4)
162 $ff5e -$00a2 MoveScreen(screen,dx,dy)(a0,d0/d1)
168 $ff58 -$00a8 MoveWindow(window,dx,dy)(a0,d0/d1)
174 $ff52 -$00ae OffGadget(gadget,window,requester)(a0/a1/a2)
180 $ff4c -$00b4 OffMenu(window,menuNumber)(a0,d0)
186 $ff46 -$00ba OnGadget(gadget,window,requester)(a0/a1/a2)
192 $ff40 -$00c0 OnMenu(window,menuNumber)(a0,d0)
198 $ff3a -$00c6 OpenScreen(newScreen)(a0)
204 $ff34 -$00cc OpenWindow(newWindow)(a0)
```

Function Offset Reference

```
210 $ff2e -$00d2 OpenWorkBench()()
216 $ff28 -$00d8 PrintIText(rp,iText,left,top)(a0/a1,d0/d1)
222 $ff22 -$00de RefreshGadgets(gadgets,window,requester)(a0/a1/a2)
228 $ff1c -$00e4 RemoveGadget(window,gadget)(a0/a1)
* The official calling sequence for ReportMouse is given below.
* Note the register order.  For the complete story, read the ReportMouse
* autodoc.
234 $ff16 -$00ea ReportMouse(flag,window)(d0/a0)
240 $ff10 -$00f0 Request(requester,window)(a0/a1)
246 $ff0a -$00f6 ScreenToBack(screen)(a0)
252 $ff04 -$00fc ScreenToFront(screen)(a0)
258 $fefe -$0102 SetDMRequest(window,requester)(a0/a1)
264 $fef8 -$0108 SetMenuStrip(window,menu)(a0/a1)
270 $fef2 -$010e SetPointer(window,pointer,height,width,xOffset,yOffset)
                  (a0/a1,d0/d1/d2/d3)
276 $feec -$0114 SetWindowTitles(window,windowTitle,screenTitle)(a0/a1/a2)
282 $fee6 -$011a ShowTitle(screen,showIt)(a0,d0)
288 $fee0 -$0120 SizeWindow(window,dx,dy)(a0,d0/d1)
294 $feda -$0126 ViewAddress()()
300 $fed4 -$012c ViewPortAddress(window)(a0)
306 $fece -$0132 WindowToBack(window)(a0)
312 $fec8 -$0138 WindowToFront(window)(a0)
318 $fec2 -$013e WindowLimits(window,widthMin,heightMin,widthMax,heightMax)
                  (a0,d0/d1/d2/d3)
*--- start of next generation of names -------------------------------------
324 $febc -$0144 SetPrefs(preferences,size,inform)(a0,d0/d1)
*--- start of next next generation of names --------------------------------
330 $feb6 -$014a IntuiTextLength(iText)(a0)
336 $feb0 -$0150 WBenchToBack()()
342 $feaa -$0156 WBenchToFront()()
*--- start of next next next generation of names ---------------------------
348 $fea4 -$015c AutoRequest(window,body,posText,negText,pFlag,nFlag,width,
                 height)(a0/a1/a2/a3,d0/d1/d2/d3)
354 $fe9e -$0162 BeginRefresh(window)(a0)
360 $fe98 -$0168 BuildSysRequest(window,body,posText,negText,flags,width,
                 height)(a0/a1/a2/a3,d0/d1/d2)
366 $fe92 -$016e EndRefresh(window,complete)(a0,d0)
372 $fe8c -$0174 FreeSysRequest(window)(a0)
378 $fe86 -$017a MakeScreen(screen)(a0)
384 $fe80 -$0180 RemakeDisplay()()
390 $fe7a -$0186 RethinkDisplay()()
*--- start of next next next next generation of names ----------------------
396 $fe74 -$018c AllocRemember(rememberKey,size,flags)(a0,d0/d1)
* Public function AlohaWorkbench() is intentionally not documented
402 $fe6e -$0192 AlohaWorkbench(wbport)(a0)
408 $fe68 -$0198 FreeRemember(rememberKey,reallyForget)(a0,d0)
*--- start of 15 Nov 85 names ------------------------
414 $fe62 -$019e LockIBase(dontknow)(d0)
420 $fe5c -$01a4 UnlockIBase(ibLock)(a0)
*--- functions in V33 or higher (distributed as Release 1.2) ---
426 $fe56 -$01aa GetScreenData(buffer,size,type,screen)(a0,d0/d1/a1)
432 $fe50 -$01b0 RefreshGList(gadgets,window,requester,numGad)(a0/a1/a2,d0)
438 $fe4a -$01b6 AddGList(window,gadget,position,numGad,requester)
                  (a0/a1,d0/d1/a2)
444 $fe44 -$01bc RemoveGList(remPtr,gadget,numGad)(a0/a1,d0)
450 $fe3e -$01c2 ActivateWindow(window)(a0)
456 $fe38 -$01c8 RefreshWindowFrame(window)(a0)
462 $fe32 -$01ce ActivateGadget(gadgets,window,requester)(a0/a1/a2)
468 $fe2c -$01d4 NewModifyProp(gadget,window,requester,flags,horizPot,
                 vertPot,horizBody,vertBody,numGad)(a0/a1/a2,d0/d1/d2/d3/d4/d5)
*--- functions in V36 or higher (distributed as Release 2.0) ---
474 $fe26 -$01da QueryOverscan(displayID,rect,oScanType)(a0/a1,d0)
480 $fe20 -$01e0 MoveWindowInFrontOf(window,behindWindow)(a0/a1)
486 $fe1a -$01e6 ChangeWindowBox(window,left,top,width,height)
                  (a0,d0/d1/d2/d3)
492 $fe14 -$01ec SetEditHook(hook)(a0)
```

Function Offset Reference

```
498 $fe0e -$01f2 SetMouseQueue(window,queueLength)(a0,d0)
504 $fe08 -$01f8 ZipWindow(window)(a0)
*--- public screens ---
510 $fe02 -$01fe LockPubScreen(name)(a0)
516 $fdfc -$0204 UnlockPubScreen(name,screen)(a0/a1)
522 $fdf6 -$020a LockPubScreenList()()
528 $fdf0 -$0210 UnlockPubScreenList()()
534 $fdea -$0216 NextPubScreen(screen,namebuf)(a0/a1)
540 $fde4 -$021c SetDefaultPubScreen(name)(a0)
546 $fdde -$0222 SetPubScreenModes(modes)(d0)
552 $fdd8 -$0228 PubScreenStatus(screen,statusFlags)(a0,d0)
*
558 $fdd2 -$022e ObtainGIRPort(gInfo)(a0)
564 $fdcc -$0234 ReleaseGIRPort(rp)(a0)
570 $fdc6 -$023a GadgetMouse(gadget,gInfo,mousePoint)(a0/a1/a2)
* SetIPrefs is system private and not to be used by applications
##private
576 $fdc0 -$0240 SetIPrefs(ptr,size,type)(a0,d0/d1)
##public
582 $fdba -$0246 GetDefaultPubScreen(nameBuffer)(a0)
588 $fdb4 -$024c EasyRequestArgs(window,easyStruct,idcmpPtr,args)
                 (a0/a1/a2/a3)
594 $fdae -$0252 BuildEasyRequestArgs(window,easyStruct,idcmp,args)
                  (a0/a1,d0/a3)
600 $fda8 -$0258 SysReqHandler(window,idcmpPtr,waitInput)(a0/a1,d0)
606 $fda2 -$025e OpenWindowTagList(newWindow,tagList)(a0/a1)
612 $fd9c -$0264 OpenScreenTagList(newScreen,tagList)(a0/a1)
*
*       new Image functions
618 $fd96 -$026a DrawImageState(rp,image,leftOffset,topOffset,state,drawInfo)
                 (a0/a1,d0/d1/d2/a2)
624 $fd90 -$0270 PointInImage(point,image)(d0/a0)
630 $fd8a -$0276 EraseImage(rp,image,leftOffset,topOffset)(a0/a1,d0/d1)
*
636 $fd84 -$027c NewObjectA(class,classID,tagList)(a0/a1/a2)
*
642 $fd7e -$0282 DisposeObject(object)(a0)
648 $fd78 -$0288 SetAttrsA(object,tagList)(a0/a1)
*
654 $fd72 -$028e GetAttr(attrID,object,storagePtr)(d0/a0/a1)
*
*       special set attribute call for gadgets
660 $fd6c -$0294 SetGadgetAttrsA(gadget,window,requester,tagList)
                  (a0/a1/a2/a3)
*
*       for class implementors only
666 $fd66 -$029a NextObject(objectPtrPtr)(a0)
##private
672 $fd60 -$02a0 FindClass(classID)(a0)
##public
678 $fd5a -$02a6 MakeClass(classID,superClassID,superClassPtr,instanceSize,
                  flags)(a0/a1/a2,d0/d1)
684 $fd54 -$02ac AddClass(class)(a0)
*
*
690 $fd4e -$02b2 GetScreenDrawInfo(screen)(a0)
696 $fd48 -$02b8 FreeScreenDrawInfo(screen,drawInfo)(a0/a1)
*
702 $fd42 -$02be ResetMenuStrip(window,menu)(a0/a1)
708 $fd3c -$02c4 RemoveClass(classPtr)(a0)
714 $fd36 -$02ca FreeClass(classPtr)(a0)
##private
720 $fd30 -$02d0 lockPubClass()()
726 $fd2a -$02d6 unlockPubClass()()
##public
##end
```

Function Offset Reference Page 17

```
***************** keymap
* "keymap.library"
##base _KeymapBase
##bias 30
##public
*--- functions in V36 or higher (distributed as Release 2.0) ---
 30 $ffe2 -$001e SetKeyMapDefault(keyMap)(a0)
 36 $ffdc -$0024 AskKeyMapDefault()()
 42 $ffd6 -$002a MapRawKey(event,buffer,length,keyMap)(a0/a1,d1/a2)
 48 $ffd0 -$0030 MapANSI(string,count,buffer,length,keyMap)(a0,d0/a1,d1/a2)
##end

***************** layers
* "layers.library"
##base _LayersBase
##bias 30
##public
 30 $ffe2 -$001e InitLayers(li)(a0)
 36 $ffdc -$0024 CreateUpfrontLayer(li,bm,x0,y0,x1,y1,flags,bm2)
                  (a0/a1,d0/d1/d2/d3/d4/a2)
 42 $ffd6 -$002a CreateBehindLayer(li,bm,x0,y0,x1,y1,flags,bm2)
                  (a0/a1,d0/d1/d2/d3/d4/a2)
 48 $ffd0 -$0030 UpfrontLayer(dummy,layer)(a0/a1)
 54 $ffca -$0036 BehindLayer(dummy,layer)(a0/a1)
 60 $ffc4 -$003c MoveLayer(dummy,layer,dx,dy)(a0/a1,d0/d1)
 66 $ffbe -$0042 SizeLayer(dummy,layer,dx,dy)(a0/a1,d0/d1)
 72 $ffb8 -$0048 ScrollLayer(dummy,layer,dx,dy)(a0/a1,d0/d1)
 78 $ffb2 -$004e BeginUpdate(l)(a0)
 84 $ffac -$0054 EndUpdate(layer,flag)(a0,d0)
 90 $ffa6 -$005a DeleteLayer(dummy,layer)(a0/a1)
 96 $ffa0 -$0060 LockLayer(dummy,layer)(a0/a1)
102 $ff9a -$0066 UnlockLayer(layer)(a0)
108 $ff94 -$006c LockLayers(li)(a0)
114 $ff8e -$0072 UnlockLayers(li)(a0)
120 $ff88 -$0078 LockLayerInfo(li)(a0)
126 $ff82 -$007e SwapBitsRastPortClipRect(rp,cr)(a0/a1)
132 $ff7c -$0084 WhichLayer(li,x,y)(a0,d0/d1)
138 $ff76 -$008a UnlockLayerInfo(li)(a0)
144 $ff70 -$0090 NewLayerInfo()()
150 $ff6a -$0096 DisposeLayerInfo(li)(a0)
156 $ff64 -$009c FattenLayerInfo(li)(a0)
162 $ff5e -$00a2 ThinLayerInfo(li)(a0)
168 $ff58 -$00a8 MoveLayerInFrontOf(layer_to_move,other_layer)(a0/a1)
174 $ff52 -$00ae InstallClipRegion(layer,region)(a0/a1)
180 $ff4c -$00b4 MoveSizeLayer(layer,dx,dy,dw,dh)(a0,d0/d1/d2/d3)
186 $ff46 -$00ba CreateUpfrontHookLayer(li,bm,x0,y0,x1,y1,flags,hook,bm2)
                  (a0/a1,d0/d1/d2/d3/d4/a3,a2)
192 $ff40 -$00c0 CreateBehindHookLayer(li,bm,x0,y0,x1,y1,flags,hook,bm2)
                  (a0/a1,d0/d1/d2/d3/d4/a3,a2)
198 $ff3a -$00c6 InstallLayerHook(layer,hook)(a0/a1)
##end

***************** mathffp
* "mathffp.library"
##base _MathBase
##bias 30
##public
 30 $ffe2 -$001e SPFix(parm)(d0)
 36 $ffdc -$0024 SPFlt(integer)(d0)
 42 $ffd6 -$002a SPCmp(leftParm,rightParm)(d1,d0)
 48 $ffd0 -$0030 SPTst(parm)(d1)
 54 $ffca -$0036 SPAbs(parm)(d0)
 60 $ffc4 -$003c SPNeg(parm)(d0)
 66 $ffbe -$0042 SPAdd(leftParm,rightParm)(d1,d0)
 72 $ffb8 -$0048 SPSub(leftParm,rightParm)(d1,d0)
```

Function Offset Reference Page 18

```
 78 $ffb2 -$004e SPMul(leftParm,rightParm)(d1,d0)
 84 $ffac -$0054 SPDiv(leftParm,rightParm)(d1,d0)
*--- functions in V33 or higher (distributed as Release 1.2) ---
 90 $ffa6 -$005a SPFloor(parm)(d0)
 96 $ffa0 -$0060 SPCeil(parm)(d0)
##end

***************** mathieeedoubbas
* "mathieeedoubbas.library"
##base _MathIeeeDoubBasBase
##bias 30
##public
 30 $ffe2 -$001e IEEEDPFix(parm)(d0/d1)
 36 $ffdc -$0024 IEEEDPFlt(integer)(d0)
 42 $ffd6 -$002a IEEEDPCmp(leftParm,rightParm)(d0/d1/d2/d3)
 48 $ffd0 -$0030 IEEEDPTst(parm)(d0/d1)
 54 $ffca -$0036 IEEEDPAbs(parm)(d0/d1)
 60 $ffc4 -$003c IEEEDPNeg(parm)(d0/d1)
 66 $ffbe -$0042 IEEEDPAdd(leftParm,rightParm)(d0/d1/d2/d3)
 72 $ffb8 -$0048 IEEEDPSub(leftParm,rightParm)(d0/d1/d2/d3)
 78 $ffb2 -$004e IEEEDPMul(factor1,factor2)(d0/d1/d2/d3)
 84 $ffac -$0054 IEEEDPDiv(dividend,divisor)(d0/d1/d2/d3)
*--- functions in V33 or higher (distributed as Release 1.2) ---
 90 $ffa6 -$005a IEEEDPFloor(parm)(d0/d1)
 96 $ffa0 -$0060 IEEEDPCeil(parm)(d0/d1)
##end

***************** mathieeedoubtrans
* "mathieeedoubtrans.library"
##base _MathIeeeDoubTransBase
##bias 30
##public
 30 $ffe2 -$001e IEEEDPAtan(parm)(d0/d1)
 36 $ffdc -$0024 IEEEDPSin(parm)(d0/d1)
 42 $ffd6 -$002a IEEEDPCos(parm)(d0/d1)
 48 $ffd0 -$0030 IEEEDPTan(parm)(d0/d1)
 54 $ffca -$0036 IEEEDPSincos(pf2,parm)(a0,d0/d1)
 60 $ffc4 -$003c IEEEDPSinh(parm)(d0/d1)
 66 $ffbe -$0042 IEEEDPCosh(parm)(d0/d1)
 72 $ffb8 -$0048 IEEEDPTanh(parm)(d0/d1)
 78 $ffb2 -$004e IEEEDPExp(parm)(d0/d1)
 84 $ffac -$0054 IEEEDPLog(parm)(d0/d1)
 90 $ffa6 -$005a IEEEDPPow(exp,arg)(d2/d3,d0/d1)
 96 $ffa0 -$0060 IEEEDPSqrt(parm)(d0/d1)
102 $ff9a -$0066 IEEEDPTieee(parm)(d0/d1)
108 $ff94 -$006c IEEEDPFieee(single)(d0)
114 $ff8e -$0072 IEEEDPAsin(parm)(d0/d1)
120 $ff88 -$0078 IEEEDPAcos(parm)(d0/d1)
126 $ff82 -$007e IEEEDPLog10(parm)(d0/d1)
##end

***************** mathieeesingbas
* "mathieeesingbas.library"
##base _MathIeeeSingBasBase
##bias 30
##public
 30 $ffe2 -$001e IEEESPFix(parm)(d0)
 36 $ffdc -$0024 IEEESPFlt(integer)(d0)
 42 $ffd6 -$002a IEEESPCmp(leftParm,rightParm)(d0/d1)
 48 $ffd0 -$0030 IEEESPTst(parm)(d0)
 54 $ffca -$0036 IEEESPAbs(parm)(d0)
 60 $ffc4 -$003c IEEESPNeg(parm)(d0)
 66 $ffbe -$0042 IEEESPAdd(leftParm,rightParm)(d0/d1)
 72 $ffb8 -$0048 IEEESPSub(leftParm,rightParm)(d0/d1)
 78 $ffb2 -$004e IEEESPMul(leftParm,rightParm)(d0/d1)
 84 $ffac -$0054 IEEESPDiv(dividend,divisor)(d0/d1)
 90 $ffa6 -$005a IEEESPFloor(parm)(d0)
 96 $ffa0 -$0060 IEEESPCeil(parm)(d0)
##end

***************** mathieeesingtrans
* "mathieeesingtrans.library"
##base _MathIeeeSingTransBase
##bias 30
##public
 30 $ffe2 -$001e IEEESPAtan(parm)(d0)
 36 $ffdc -$0024 IEEESPSin(parm)(d0)
 42 $ffd6 -$002a IEEESPCos(parm)(d0)
 48 $ffd0 -$0030 IEEESPTan(parm)(d0)
 54 $ffca -$0036 IEEESPSincos(cosptr,parm)(a0,d0)
 60 $ffc4 -$003c IEEESPSinh(parm)(d0)
 66 $ffbe -$0042 IEEESPCosh(parm)(d0)
 72 $ffb8 -$0048 IEEESPTanh(parm)(d0)
 78 $ffb2 -$004e IEEESPExp(parm)(d0)
 84 $ffac -$0054 IEEESPLog(parm)(d0)
 90 $ffa6 -$005a IEEESPPow(exp,arg)(d1,d0)
 96 $ffa0 -$0060 IEEESPSqrt(parm)(d0)
102 $ff9a -$0066 IEEESPTieee(parm)(d0)
108 $ff94 -$006c IEEESPFieee(parm)(d0)
114 $ff8e -$0072 IEEESPAsin(parm)(d0)
120 $ff88 -$0078 IEEESPAcos(parm)(d0)
126 $ff82 -$007e IEEESPLog10(parm)(d0)
##end

***************** mathtrans
* "mathtrans.library"
##base _MathTransBase
##bias 30
##public
 30 $ffe2 -$001e SPAtan(parm)(d0)
 36 $ffdc -$0024 SPSin(parm)(d0)
 42 $ffd6 -$002a SPCos(parm)(d0)
 48 $ffd0 -$0030 SPTan(parm)(d0)
 54 $ffca -$0036 SPSincos(cosResult,parm)(d1,d0)
 60 $ffc4 -$003c SPSinh(parm)(d0)
 66 $ffbe -$0042 SPCosh(parm)(d0)
 72 $ffb8 -$0048 SPTanh(parm)(d0)
 78 $ffb2 -$004e SPExp(parm)(d0)
 84 $ffac -$0054 SPLog(parm)(d0)
 90 $ffa6 -$005a SPPow(power,arg)(d1,d0)
 96 $ffa0 -$0060 SPSqrt(parm)(d0)
102 $ff9a -$0066 SPTieee(parm)(d0)
108 $ff94 -$006c SPFieee(parm)(d0)
*--- functions in V31 or higher (distributed as Release 1.1) ---
114 $ff8e -$0072 SPAsin(parm)(d0)
120 $ff88 -$0078 SPAcos(parm)(d0)
126 $ff82 -$007e SPLog10(parm)(d0)
##end

***************** misc
##base _MiscBase
##bias 6
##public
  6 $fffa -$0006 AllocMiscResource(unitNum,name)(d0/a1)
 12 $fff4 -$000c FreeMiscResource(unitNum)(d0)
##end

***************** potgo
* "potgo.resource"
##base _PotgoBase
##bias 6
##public
```

```
  6 $fffa -$0006 AllocPotBits(bits)(d0)
 12 $fff4 -$000c FreePotBits(bits)(d0)
 18 $ffee -$0012 WritePotgo(word,mask)(d0/d1)
##end

***************** ramdrive
* "ramdrive.device"
##base _RamdriveDevice
##bias 42
##public
*--- functions in V34 or higher (distributed as Release 1.3) ---
 42 $ffd6 -$002a KillRAD0()()
*--- functions in V36 or higher (distributed as Release 2.0) ---
 48 $ffd0 -$0030 KillRAD(unit)(d0)
##end

***************** rexxsyslib
* "rexxsyslib.library"
##base _RexxSysBase
##bias 30
*--- functions in V33 or higher (distributed as Release 1.2) ---
*
##private
*--- (16 function slots reserved here) ---
##bias 126
##public
*
126 $ff82 -$007e CreateArgstring(string,length)(a0,d0)
132 $ff7c -$0084 DeleteArgstring(argstring)(a0)
138 $ff76 -$008a LengthArgstring(argstring)(a0)
144 $ff70 -$0090 CreateRexxMsg(port,extension,host)(a0/a1,d0)
150 $ff6a -$0096 DeleteRexxMsg(packet)(a0)
156 $ff64 -$009c ClearRexxMsg(msgptr,count)(a0,d0)
162 $ff5e -$00a2 FillRexxMsg(msgptr,count,mask)(a0,d0/d1)
168 $ff58 -$00a8 IsRexxMsg(msgptr)(a0)
*
##private
*--- (46 function slots reserved here) ---
##bias 450
##public
*
450 $fe3e -$01c2 LockRexxBase(resource)(d0)
456 $fe38 -$01c8 UnlockRexxBase(resource)(d0)
*
##end

***************** timer
* "Timer.Device"
##base _TimerBase
##bias 42
##public
 42 $ffd6 -$002a AddTime(dest,src)(a0/a1)
 48 $ffd0 -$0030 SubTime(dest,src)(a0/a1)
 54 $ffca -$0036 CmpTime(dest,src)(a0/a1)
 60 $ffc4 -$003c ReadEClock(dest)(a0)
 66 $ffbe -$0042 GetSysTime(dest)(a0)
##end

***************** translator
* "translator.library"
##base _TranslatorBase
##bias 30
##public
 30 $ffe2 -$001e Translate(inputString,inputLength,outputBuffer,bufferSize)
                 (a0,d0/a1,d1)
##end
```

Function Offset Reference Page 21

```
***************** utility
* "utility.library"
##base _UtilityBase
##bias 30
##public
* *** TagItem FUNCTIONS ***
 30 $ffe2 -$001e FindTagItem(tagVal,tagList)(d0/a0)
 36 $ffdc -$0024 GetTagData(tagVal,defaultVal,tagList)(d0/d1/a0)
 42 $ffd6 -$002a PackBoolTags(initialFlags,tagList,boolMap)(d0/a0/a1)
 48 $ffd0 -$0030 NextTagItem(tagListPtr)(a0)
 54 $ffca -$0036 FilterTagChanges(newTagList,oldTagList,apply)(a0/a1,d0)
 60 $ffc4 -$003c MapTags(tagList,mapList,includeMiss)(a0/a1,d0)
 66 $ffbe -$0042 AllocateTagItems(numItems)(d0)
 72 $ffb8 -$0048 CloneTagItems(tagList)(a0)
 78 $ffb2 -$004e FreeTagItems(tagList)(a0)
 84 $ffac -$0054 RefreshTagItemClones(cloneList,origList)(a0/a1)
 90 $ffa6 -$005a TagInArray(tagVal,tagArray)(d0/a0)
 96 $ffa0 -$0060 FilterTagItems(tagList,filterArray,logic)(a0/a1,d0)
*
* *** HOOK FUNCTIONS *** *
102 $ff9a -$0066 CallHookPkt(hook,object,paramPacket)(a0/a2,a1)
*--- (1 function slot reserved here) ---
##bias 114
*
* *** DATE FUNCTIONS *** *
*--- (1 function slot reserved here) ---
##bias 120
120 $ff88 -$0078 Amiga2Date(amigaTime,date)(d0/a0)
126 $ff82 -$007e Date2Amiga(date)(a0)
132 $ff7c -$0084 CheckDate(date)(a0)
*
* *** 32 BIT MATH FUNCTIONS *** *
138 $ff76 -$008a SMult32(factor1,factor2)(d0/d1)
144 $ff70 -$0090 UMult32(factor1,factor2)(d0/d1)
* NOTE: Quotient:Remainder returned in d0:d1
150 $ff6a -$0096 SDivMod32(dividend,divisor)(d0/d1)
156 $ff64 -$009c UDivMod32(dividend,divisor)(d0/d1)
*
* *** International string routines ***
162 $ff5e -$00a2 Stricmp(string1,string2)(a0/a1)
168 $ff58 -$00a8 Strnicmp(string1,string2,length)(a0/a1,d0)
174 $ff52 -$00ae ToUpper(character)(d0)
180 $ff4c -$00b4 ToLower(character)(d0)
##end

***************** wb
* "workbench.library"
##base _WorkbenchBase
##bias 30
*--- functions in V36 or higher (distributed as Release 2.0) ---
*
##private
*
 30 $ffe2 -$001e UpdateWorkbench(name,lock,flag)(a0/a1,d0)
 36 $ffdc -$0024 QuoteWorkbench(stringNum)(d0)
*
 42 $ffd6 -$002a StartWorkbench(flags,ptr)(d0/d1)
*
##public
*
 48 $ffd0 -$0030 AddAppWindowA(id,userdata,window,msgport,taglist)
                 (d0/d1/a0/a1/a2)
*
 54 $ffca -$0036 RemoveAppWindow(appWindow)(a0)
*
```

Function Offset Reference Page 22

```
 60 $ffc4 -$003c AddAppIconA(id,userdata,text,msgport,lock,diskobj,taglist)
                 (d0/d1/a0/a1/a2/a3/a4)
*
 66 $ffbe -$0042 RemoveAppIcon(appIcon)(a0)
*
 72 $ffb8 -$0048 AddAppMenuItemA(id,userdata,text,msgport,taglist)
                 (d0/d1/a0/a1/a2)
*
 78 $ffb2 -$004e RemoveAppMenuItem(appMenuItem)(a0)
*
*--- (5 function slots reserved here) ---
##bias 114
*
##end
```

Structure Reference Page 1

```
AChain:
  $0112  274  sizeof(AChain)
  $0000    0  an_Child
  $0004    4  an_Parent
  $0008    8  an_Lock
  $000c   12  an_Info
  $0110  272  an_Flags
  $0111  273  an_String[0]
AnalogSignalInterval:
  $0004    4  sizeof(
         AnalogSignalInterval)
  $0000    0  asi_Start
  $0002    2  asi_Stop
AnchorPath:
  $011a  282  sizeof(AnchorPath)
  $0000    0  ap_Base
  $0004    4  ap_Last
  $0008    8  ap_BreakBits
  $000c   12  ap_FoundBreak
  $0010   16  ap_Flags
  $0011   17  ap_Reserved
  $0012   18  ap_Strlen
  $0014   20  ap_Info
  $0118  280  ap_Buf[0]
AnimComp:
  $0026   38  sizeof(AnimComp)
  $0000    0  Flags
  $0002    2  Timer
  $0004    4  TimeSet
! $0006    6  NextComp
! $000a   10  PrevComp
! $000e   14  NextSeq
! $0012   18  PrevSeq
! $0016   22  AnimCRoutine
  $001a   26  YTrans
  $001c   28  XTrans
! $001e   30  HeadOb
! $0022   34  AnimBob
AnimOb:
  $002a   42  sizeof(AnimOb)
  $0000    0  NextOb
  $0004    4  PrevOb
  $0008    8  Clock
  $000c   12  AnOldY
  $000e   14  AnOldX
  $0010   16  AnY
  $0012   18  AnX
  $0014   20  YVel
  $0016   22  XVel
  $0018   24  YAccel
  $001a   26  XAccel
  $001c   28  RingYTrans
  $001e   30  RingXTrans
  $0020   32  AnimORoutine
  $0024   36  HeadComp
  $0028   40  AUserExt
AppIcon:
  $0020   32  sizeof(AppIcon)
  $0000    0  ai_Node
! $000e   14  ai_Ptr
! $0012   18  ai_MsgPort
! $0016   22  ai_UserData
! $001a   26  ai_ID
  $001e   30  ai_Version
AppMenuItem:
  $0020   32  sizeof(AppMenuItem)
```

```
  $0000    0  ami_Node
! $000e   14  ami_Ptr
! $0012   18  ami_MsgPort
! $0016   22  ami_UserData
! $001a   26  ami_ID
  $001e   30  ami_Version
AppMessage:
  $0056   86  sizeof(AppMessage)
  $0000    0  am_Message
  $0014   20  am_Type
! $0016   22  am_UserData
! $001a   26  am_ID
! $001e   30  am_NumArgs
! $0022   34  am_ArgList
  $0026   38  am_Version
  $0028   40  am_Class
  $002a   42  am_MouseX
  $002c   44  am_MouseY
! $002e   46  am_Seconds
! $0032   50  am_Micros
! $0036   54  am_Reserved[0]
AppWindow:
  $0020   32  sizeof(AppWindow)
  $0000    0  aw_Node
! $000e   14  aw_Ptr
! $0012   18  aw_MsgPort
! $0016   22  aw_UserData
! $001a   26  aw_ID
  $001e   30  aw_Version
AreaInfo:
  $0018   24  sizeof(AreaInfo)
  $0000    0  VctrTbl
  $0004    4  VctrPtr
  $0008    8  FlagTbl
  $000c   12  FlagPtr
  $0010   16  Count
  $0012   18  MaxCount
  $0014   20  FirstX
  $0016   22  FirstY
AssignList:
  $0008    8  sizeof(AssignList)
  $0000    0  al_Next
  $0004    4  al_Lock
AudChannel:
  $0010   16  sizeof(AudChannel)
  $0000    0  ac_ptr
  $0004    4  ac_len
  $0006    6  ac_per
  $0008    8  ac_vol
  $000a   10  ac_dat
  $000c   12  ac_pad[0]
AvailFonts:
  $000a   10  sizeof(AvailFonts)
  $0000    0  af_Type
! $0002    2  af_Attr
AvailFontsHeader:
  $0002    2  sizeof(AvailFontsHeader)
  $0000    0  afh_NumEntries
BadBlockBlock:
  $0200  512  sizeof(BadBlockBlock)
  $0000    0  bbb_ID
  $0004    4  bbb_SummedLongs
  $0008    8  bbb_ChkSum
  $000c   12  bbb_HostID
  $0010   16  bbb_Next
  $0014   20  bbb_Reserved
```

Structure Reference Page 2

```
  $0018   24  bbb_BlockPairs[0]
BadBlockEntry:
  $0008    8  sizeof(BadBlockEntry)
  $0000    0  bbe_BadBlock
  $0004    4  bbe_GoodBlock
BitMap:
  $0028   40  sizeof(BitMap)
  $0000    0  BytesPerRow
  $0002    2  Rows
  $0004    4  Flags
  $0005    5  Depth
  $0006    6  pad
  $0008    8  Planes[0]
BitScaleArgs:
  $0030   48  sizeof(BitScaleArgs)
  $0000    0  bsa_SrcX
  $0002    2  bsa_SrcY
  $0004    4  bsa_SrcWidth
  $0006    6  bsa_SrcHeight
  $0008    8  bsa_XSrcFactor
  $000a   10  bsa_YSrcFactor
  $000c   12  bsa_DestX
  $000e   14  bsa_DestY
  $0010   16  bsa_DestWidth
  $0012   18  bsa_DestHeight
  $0014   20  bsa_XDestFactor
  $0016   22  bsa_YDestFactor
  $0018   24  bsa_SrcBitMap
  $001c   28  bsa_DestBitMap
  $0020   32  bsa_Flags
  $0024   36  bsa_XDDA
  $0026   38  bsa_YDDA
  $0028   40  bsa_Reserved1
  $002c   44  bsa_Reserved2
Bob:
  $0020   32  sizeof(Bob)
  $0000    0  Flags
! $0002    2  SaveBuffer
! $0006    6  ImageShadow
! $000a   10  Before
! $000e   14  After
! $0012   18  BobVSprite
! $0016   22  BobComp
! $001a   26  DBuffer
  $001e   30  BUserExt
BoolInfo:
  $000a   10  sizeof(BoolInfo)
  $0000    0  Flags
! $0002    2  Mask
! $0006    6  Reserved
BootBlock:
  $000c   12  sizeof(BootBlock)
  $0000    0  bb_id[0]
  $0004    4  bb_chksum
  $0008    8  bb_dosblock
BootNode:
  $0014   20  sizeof(BootNode)
  $0000    0  bn_Node
  $000e   14  bn_Flags
  $0010   16  bn_DeviceNode
Border:
  $0010   16  sizeof(Border)
  $0000    0  LeftEdge
  $0002    2  TopEdge
  $0004    4  FrontPen
  $0005    5  BackPen
```

```
  $0006    6  DrawMode
  $0007    7  Count
  $0008    8  XY
  $000c   12  NextBorder
CIA:
  $0f02 3842  sizeof(CIA)
  $0000    0  ciapra
! $0001    1  pad0[0]
  $0100  256  ciaprb
! $0101  257  pad1[0]
  $0200  512  ciaddra
! $0201  513  pad2[0]
  $0300  768  ciaddrb
! $0301  769  pad3[0]
  $0400 1024  ciatalo
! $0401 1025  pad4[0]
  $0500 1280  ciatahi
! $0501 1281  pad5[0]
  $0600 1536  ciatblo
! $0601 1537  pad6[0]
  $0700 1792  ciatbhi
! $0701 1793  pad7[0]
  $0800 2048  ciatodlow
! $0801 2049  pad8[0]
  $0900 2304  ciatodmid
! $0901 2305  pad9[0]
  $0a00 2560  ciatodhi
! $0a01 2561  pad10[0]
  $0b00 2816  unusedreg
! $0b01 2817  pad11[0]
  $0c00 3072  ciasdr
! $0c01 3073  pad12[0]
  $0d00 3328  ciaicr
! $0d01 3329  pad13[0]
  $0e00 3584  ciacra
! $0e01 3585  pad14[0]
  $0f00 3840  ciacrb
CSource:
  $000c   12  sizeof(CSource)
  $0000    0  CS_Buffer
  $0004    4  CS_Length
  $0008    8  CS_CurChr
CliProcList:
  $0010   16  sizeof(CliProcList)
  $0000    0  cpl_Node
  $0008    8  cpl_First
  $000c   12  cpl_Array
ClipHookMsg:
  $000c   12  sizeof(ClipHookMsg)
  $0000    0  chm_Type
  $0004    4  chm_ChangeCmd
  $0008    8  chm_ClipID
ClipRect:
  $0024   36  sizeof(ClipRect)
  $0000    0  Next
  $0004    4  prev
  $0008    8  lobs
  $000c   12  BitMap
  $0010   16  bounds
  $0018   24  _p1
  $001c   28  _p2
  $0020   32  reserved
ClipboardHandle:
  $0078  120  sizeof(ClipboardHandle)
  $0000    0  cbh_Req
  $0034   52  cbh_CBport
```

Structure Reference Page 3

```
! $0056    86  cbh_SatisfyPort
ClipboardUnitPartial:
  $0012    18  sizeof(
        ClipboardUnitPartial)
  $0000     0  cu_Node
! $000e    14  cu_UnitNum
ClockData:
  $000e    14  sizeof(ClockData)
  $0000     0  sec
  $0002     2  min
  $0004     4  hour
  $0006     6  mday
  $0008     8  month
  $000a    10  year
  $000c    12  wday
CollectionItem:
  $000c    12  sizeof(CollectionItem)
  $0000     0  ci_Next
  $0004     4  ci_Size
  $0008     8  ci_Data
ColorFontColors:
  $0008     8  sizeof(ColorFontColors)
  $0000     0  cfc_Reserved
  $0002     2  cfc_Count
  $0004     4  cfc_ColorTable
ColorMap:
  $0028    40  sizeof(ColorMap)
  $0000     0  Flags
  $0001     1  Type
  $0002     2  Count
  $0004     4  ColorTable
  $0008     8  cm_vpe
  $000c    12  TransparencyBits
  $0010    16  TransparencyPlane
  $0011    17  reserved1
  $0012    18  reserved2
  $0014    20  cm_vp
  $0018    24  NormalDisplayInfo
  $001c    28  CoerceDisplayInfo
  $0020    32  cm_batch_items
  $0024    36  VPModeID
ColorSpec:
  $0008     8  sizeof(ColorSpec)
  $0000     0  ColorIndex
  $0002     2  Red
  $0004     4  Green
  $0006     6  Blue
ColorTextFont:
  $0060    96  sizeof(ColorTextFont)
  $0000     0  ctf_TF
  $0034    52  ctf_Flags
  $0036    54  ctf_Depth
  $0037    55  ctf_FgColor
  $0038    56  ctf_Low
  $0039    57  ctf_High
  $003a    58  ctf_PlanePick
  $003b    59  ctf_PlaneOnOff
  $003c    60  ctf_ColorFontColors
  $0040    64  ctf_CharData[0]
CommandLineInterface:
  $0040    64  sizeof(
        CommandLineInterface)
  $0000     0  cli_Result2
  $0004     4  cli_SetName
  $0008     8  cli_CommandDir
  $000c    12  cli_ReturnCode
```

```
  $0010    16  cli_CommandName
  $0014    20  cli_FailLevel
  $0018    24  cli_Prompt
  $001c    28  cli_StandardInput
  $0020    32  cli_CurrentInput
  $0024    36  cli_CommandFile
  $0028    40  cli_Interactive
  $002c    44  cli_Background
  $0030    48  cli_CurrentOutput
  $0034    52  cli_DefaultStack
  $0038    56  cli_StandardOutput
  $003c    60  cli_Module
ConUnit:
  $0128   296  sizeof(ConUnit)
  $0000     0  cu_MP
! $0022    34  cu_Window
  $0026    38  cu_XCP
  $0028    40  cu_YCP
  $002a    42  cu_XMax
  $002c    44  cu_YMax
  $002e    46  cu_XRSize
  $0030    48  cu_YRSize
  $0032    50  cu_XROrigin
  $0034    52  cu_YROrigin
  $0036    54  cu_XRExtant
  $0038    56  cu_YRExtant
  $003a    58  cu_XMinShrink
  $003c    60  cu_YMinShrink
  $003e    62  cu_XCCP
  $0040    64  cu_YCCP
! $0042    66  cu_KeyMapStruct
! $0062    98  cu_TabStops[0]
  $0102   258  cu_Mask
  $0103   259  cu_FgPen
  $0104   260  cu_BgPen
  $0105   261  cu_AOLPen
  $0106   262  cu_DrawMode
  $0107   263  cu_Obsolete1
  $0108   264  cu_Obsolete2
  $010c   268  cu_Minterms[0]
  $0114   276  cu_Font
  $0118   280  cu_AlgoStyle
  $0119   281  cu_TxFlags
  $011a   282  cu_TxHeight
  $011c   284  cu_TxWidth
  $011e   286  cu_TxBaseline
  $0120   288  cu_TxSpacing
  $0122   290  cu_Modes[0]
  $0125   293  cu_RawEvents[0]
ConfigDev:
  $0044    68  sizeof(ConfigDev)
  $0000     0  cd_Node
  $000e    14  cd_Flags
  $000f    15  cd_Pad
  $0010    16  cd_Rom
  $0020    32  cd_BoardAddr
  $0024    36  cd_BoardSize
  $0028    40  cd_SlotAddr
  $002a    42  cd_SlotSize
  $002c    44  cd_Driver
  $0030    48  cd_NextCD
  $0034    52  cd_Unused[0]
ContextNode:
  $0018    24  sizeof(ContextNode)
  $0000     0  cn_Node
  $0008     8  cn_ID
```

Structure Reference Page 4

```
  $000c    12  cn_Type
  $0010    16  cn_Size
  $0014    20  cn_Scan
CopIns:
  $0006     6  sizeof(CopIns)
  $0000     0  OpCode
  $0002     2  u3
  $0002     2  u3.nxtlist
  $0002     2  u3.u4
  $0002     2  u3.u4.u1
  $0002     2  u3.u4.u1.VWaitPos
  $0002     2  u3.u4.u1.DestAddr
  $0004     4  u3.u4.u2
  $0004     4  u3.u4.u2.HWaitPos
  $0004     4  u3.u4.u2.DestData
CopList:
  $0022    34  sizeof(CopList)
  $0000     0  Next
  $0004     4  _CopList
  $0008     8  _ViewPort
  $000c    12  CopIns
  $0010    16  CopPtr
  $0014    20  CopLStart
  $0018    24  CopSStart
  $001c    28  Count
  $001e    30  MaxCount
  $0020    32  DyOffset
CurrentBinding:
  $0010    16  sizeof(CurrentBinding)
  $0000     0  cb_ConfigDev
  $0004     4  cb_FileName
  $0008     8  cb_ProductString
  $000c    12  cb_ToolTypes
Custom:
  $01e6   486  sizeof(Custom)
  $0000     0  bltddat
  $0002     2  dmaconr
  $0004     4  vposr
  $0006     6  vhposr
  $0008     8  dskdatr
  $000a    10  joy0dat
  $000c    12  joy1dat
  $000e    14  clxdat
  $0010    16  adkconr
  $0012    18  pot0dat
  $0014    20  pot1dat
  $0016    22  potinp
  $0018    24  serdatr
  $001a    26  dskbytr
  $001c    28  intenar
  $001e    30  intreqr
  $0020    32  dskpt
  $0024    36  dsklen
  $0026    38  dskdat
  $0028    40  refptr
  $002a    42  vposw
  $002c    44  vhposw
  $002e    46  copcon
  $0030    48  serdat
  $0032    50  serper
  $0034    52  potgo
  $0036    54  joytest
  $0038    56  strequ
  $003a    58  strvbl
  $003c    60  strhor
  $003e    62  strlong
```

```
  $0040    64  bltcon0
  $0042    66  bltcon1
  $0044    68  bltafwm
  $0046    70  bltalwm
  $0048    72  bltcpt
  $004c    76  bltbpt
  $0050    80  bltapt
  $0054    84  bltdpt
  $0058    88  bltsize
  $005a    90  pad2d
  $005b    91  bltcon01
  $005c    92  bltsizv
  $005e    94  bltsizh
  $0060    96  bltcmod
  $0062    98  bltbmod
  $0064   100  bltamod
  $0066   102  bltdmod
  $0068   104  pad34[0]
  $0070   112  bltcdat
  $0072   114  bltbdat
  $0074   116  bltadat
! $0076   118  pad3b[0]
  $007c   124  deniseid
  $007e   126  dsksync
  $0080   128  cop1lc
  $0084   132  cop2lc
  $0088   136  copjmp1
  $008a   138  copjmp2
  $008c   140  copins
  $008e   142  diwstrt
  $0090   144  diwstop
  $0092   146  ddfstrt
  $0094   148  ddfstop
  $0096   150  dmacon
  $0098   152  clxcon
  $009a   154  intena
  $009c   156  intreq
  $009e   158  adkcon
  $0000     0  ac_ptr
  $0004     4  ac_len
  $0006     6  ac_per
  $0008     8  ac_vol
  $000a    10  ac_dat
  $000c    12  ac_pad[0]
  $00a0   160  aud[0]
  $00e0   224  bplpt[0]
  $0100   256  bplcon0
  $0102   258  bplcon1
  $0104   260  bplcon2
  $0106   262  bplcon3
  $0108   264  bpl1mod
  $010a   266  bpl2mod
  $010c   268  bplhmod
  $010e   270  pad86[0]
  $0110   272  bpldat[0]
  $0120   288  sprpt[0]
  $0000     0  pos
  $0002     2  ctl
  $0004     4  dataa
! $0006     6  datab
  $0140   320  spr[0]
  $0180   384  color[0]
  $01c0   448  htotal
  $01c2   450  hsstop
  $01c4   452  hbstrt
  $01c6   454  hbstop
  $01c8   456  vtotal
  $01ca   458  vsstop
  $01cc   460  vbstrt
  $01ce   462  vbstop
  $01d0   464  sprhstrt
  $01d2   466  sprhstop
  $01d4   468  bplhstrt
  $01d6   470  bplhstop
  $01d8   472  hhposw
  $01da   474  hhposr
  $01dc   476  beamcon0
  $01de   478  hsstrt
  $01e0   480  vsstrt
  $01e2   482  hcenter
  $01e4   484  diwhigh
DBufPacket:
  $000c    12  sizeof(DBufPacket)
  $0000     0  BufY
  $0002     2  BufX
  $0004     4  BufPath
  $0008     8  BufBuffer
DateStamp:
  $000c    12  sizeof(DateStamp)
  $0000     0  ds_Days
  $0004     4  ds_Minute
  $0008     8  ds_Tick
DateTime:
  $001a    26  sizeof(DateTime)
  $0000     0  dat_Stamp
  $000c    12  dat_Format
  $000d    13  dat_Flags
! $000e    14  dat_StrDay
! $0012    18  dat_StrDate
! $0016    22  dat_StrTime
DevInfo:
  $002c    44  sizeof(DevInfo)
  $0000     0  dvi_Next
  $0004     4  dvi_Type
  $0008     8  dvi_Task
  $000c    12  dvi_Lock
  $0010    16  dvi_Handler
  $0014    20  dvi_StackSize
  $0018    24  dvi_Priority
  $001c    28  dvi_Startup
  $0020    32  dvi_SegList
  $0024    36  dvi_GlobVec
  $0028    40  dvi_Name
DevProc:
  $0010    16  sizeof(DevProc)
  $0000     0  dvp_Port
  $0004     4  dvp_Lock
  $0008     8  dvp_Flags
  $000c    12  dvp_DevNode
Device:
  $0022    34  sizeof(Device)
  $0000     0  dd_Library
DeviceData:
  $0034    52  sizeof(DeviceData)
  $0000     0  dd_Device
! $0022    34  dd_Segment
! $0026    38  dd_ExecBase
! $002a    42  dd_CmdVectors
! $002e    46  dd_CmdBytes
  $0032    50  dd_NumCommands
DeviceList:
  $002c    44  sizeof(DeviceList)
```

```
  $0000     0  dl_Next
  $0004     4  dl_Type
  $0008     8  dl_Task
  $000c    12  dl_Lock
  $0010    16  dl_VolumeDate
  $001c    28  dl_LockList
  $0020    32  dl_DiskType
  $0024    36  dl_unused
  $0028    40  dl_Name
DeviceNode:
  $002c    44  sizeof(DeviceNode)
  $0000     0  dn_Next
  $0004     4  dn_Type
  $0008     8  dn_Task
  $000c    12  dn_Lock
  $0010    16  dn_Handler
  $0014    20  dn_StackSize
  $0018    24  dn_Priority
  $001c    28  dn_Startup
  $0020    32  dn_SegList
  $0024    36  dn_GlobalVec
  $0028    40  dn_Name
DiagArea:
  $000e    14  sizeof(DiagArea)
  $0000     0  da_Config
  $0001     1  da_Flags
  $0002     2  da_Size
  $0004     4  da_DiagPoint
  $0006     6  da_BootPoint
  $0008     8  da_Name
  $000a    10  da_Reserved01
  $000c    12  da_Reserved02
DimensionInfo:
  $0058    88  sizeof(DimensionInfo)
  $0000     0  Header
  $0010    16  MaxDepth
  $0012    18  MinRasterWidth
  $0014    20  MinRasterHeight
  $0016    22  MaxRasterWidth
  $0018    24  MaxRasterHeight
! $001a    26  Nominal
! $0022    34  MaxOScan
! $002a    42  VideoOScan
! $0032    50  TxtOScan
! $003a    58  StdOScan
! $0042    66  pad[0]
  $0050    80  reserved[0]
DiscResource:
  $0094   148  sizeof(DiscResource)
  $0000     0  dr_Library
! $0022    34  dr_Current
  $0026    38  dr_Flags
  $0027    39  dr_pad
  $0028    40  dr_SysLib
  $002c    44  dr_CiaResource
  $0030    48  dr_UnitID[0]
  $0040    64  dr_Waiting
! $004e    78  dr_DiscBlock
  $0064   100  dr_DiscSync
! $007a   122  dr_Index
  $0090   144  dr_CurrTask
DiscResourceUnit:
  $0056    86  sizeof(DiscResourceUnit)
  $0000     0  dru_Message
  $0014    20  dru_DiscBlock
! $002a    42  dru_DiscSync
```

```
  $0040    64  dru_Index
DiskFontHeader:
  $006a   106  sizeof(DiskFontHeader)
  $0000     0  dfh_DF
  $000e    14  dfh_FileID
  $0010    16  dfh_Revision
! $0012    18  dfh_Segment
! $0016    22  dfh_Name[0]
! $0036    54  dfh_TF
DiskObject:
  $004e    78  sizeof(DiskObject)
  $0000     0  do_Magic
  $0002     2  do_Version
  $0004     4  do_Gadget
  $0030    48  do_Type
! $0032    50  do_DefaultTool
! $0036    54  do_ToolTypes
! $003a    58  do_CurrentX
! $003e    62  do_CurrentY
! $0042    66  do_DrawerData
! $0046    70  do_ToolWindow
! $004a    74  do_StackSize
DisplayInfo:
  $0030    48  sizeof(DisplayInfo)
  $0000     0  Header
  $0010    16  NotAvailable
! $0012    18  PropertyFlags
! $0016    22  Resolution
  $001a    26  PixelSpeed
  $001c    28  NumStdSprites
  $001e    30  PaletteRange
  $0020    32  SpriteResolution
  $0024    36  pad[0]
  $0028    40  reserved[0]
DosEnvec:
  $0050    80  sizeof(DosEnvec)
  $0000     0  de_TableSize
  $0004     4  de_SizeBlock
  $0008     8  de_SecOrg
  $000c    12  de_Surfaces
  $0010    16  de_SectorPerBlock
  $0014    20  de_BlocksPerTrack
  $0018    24  de_Reserved
  $001c    28  de_PreAlloc
  $0020    32  de_Interleave
  $0024    36  de_LowCyl
  $0028    40  de_HighCyl
  $002c    44  de_NumBuffers
  $0030    48  de_BufMemType
  $0034    52  de_MaxTransfer
  $0038    56  de_Mask
  $003c    60  de_BootPri
  $0040    64  de_DosType
  $0044    68  de_Baud
  $0048    72  de_Control
  $004c    76  de_BootBlocks
DosInfo:
  $0070   112  sizeof(DosInfo)
  $0000     0  di_McName
  $0004     4  di_DevInfo
  $0008     8  di_Devices
  $000c    12  di_Handlers
  $0010    16  di_NetHand
  $0014    20  di_DevLock
! $0042    66  di_EntryLock
DosLibrary:
```

```
  $0042    66  sizeof(DosLibrary)
  $0000     0  dl_lib
! $0022    34  dl_Root
! $0026    38  dl_GV
! $002a    42  dl_A2
! $002e    46  dl_A5
! $0032    50  dl_A6
! $0036    54  dl_Errors
! $003a    58  dl_TimeReq
! $003e    62  dl_UtilityBase
DosList:
  $002c    44  sizeof(DosList)
  $0000     0  dol_Next
  $0004     4  dol_Type
  $0008     8  dol_Task
  $000c    12  dol_Lock
  $0010    16  dol_misc
  $0010    16  dol_misc.dol_handler
  $0010    16
        dol_misc.dol_handler.dol_Handler
  $0014    20
        dol_misc.dol_handler.dol_StackSize
  $0018    24
        dol_misc.dol_handler.dol_Priority
  $001c    28
        dol_misc.dol_handler.dol_Startup
  $0020    32
        dol_misc.dol_handler.dol_SegList
  $0024    36
        dol_misc.dol_handler.dol_GlobVec
  $0010    16  dol_misc.dol_volume
  $0010    16
        dol_misc.dol_volume.dol_VolumeDate
  $001c    28
        dol_misc.dol_volume.dol_LockList
  $0020    32
        dol_misc.dol_volume.dol_DiskType
  $0010    16  dol_misc.dol_assign
  $0010    16
        dol_misc.dol_assign.dol_AssignName
  $0014    20
        dol_misc.dol_assign.dol_List
  $0028    40  dol_Name
DosPacket:
  $0030    48  sizeof(DosPacket)
  $0000     0  dp_Link
  $0004     4  dp_Port
  $0008     8  dp_Type
  $000c    12  dp_Res1
  $0010    16  dp_Res2
  $0014    20  dp_Arg1
  $0018    24  dp_Arg2
  $001c    28  dp_Arg3
  $0020    32  dp_Arg4
  $0024    36  dp_Arg5
  $0028    40  dp_Arg6
  $002c    44  dp_Arg7
DrawInfo:
  $0032    50  sizeof(DrawInfo)
  $0000     0  dri_Version
  $0002     2  dri_NumPens
  $0004     4  dri_Pens
  $0008     8  dri_Font
  $000c    12  dri_Depth
  $000e    14  dri_Resolution
  $000e    14  dri_Resolution.X
```

Structure Reference

Page 7

```
  $0010   16  dri_Resolution.Y
! $0012   18  dri_Flags
! $0016   22  dri_Reserved[0]
DrawerData:
  $003e   62  sizeof(DrawerData)
  $0000    0  dd_NewWindow
  $0030   48  dd_CurrentX
  $0034   52  dd_CurrentY
  $0038   56  dd_Flags
  $003c   60  dd_ViewModes
DriveGeometry:
  $0020   32  sizeof(DriveGeometry)
  $0000    0  dg_SectorSize
  $0004    4  dg_TotalSectors
  $0008    8  dg_Cylinders
  $000c   12  dg_CylSectors
  $0010   16  dg_Heads
  $0014   20  dg_TrackSectors
  $0018   24  dg_BufMemType
  $001c   28  dg_DeviceType
  $001d   29  dg_Flags
  $001e   30  dg_Reserved
EClockVal:
  $0008    8  sizeof(EClockVal)
  $0000    0  ev_hi
  $0004    4  ev_lo
EasyStruct:
  $0014   20  sizeof(EasyStruct)
  $0000    0  es_StructSize
  $0004    4  es_Flags
  $0008    8  es_Title
  $000c   12  es_TextFormat
  $0010   16  es_GadgetFormat
ErrorString:
  $0008    8  sizeof(ErrorString)
  $0000    0  estr_Nums
  $0004    4  estr_Strings
ExAllControl:
  $0010   16  sizeof(ExAllControl)
  $0000    0  eac_Entries
  $0004    4  eac_LastKey
  $0008    8  eac_MatchString
  $000c   12  eac_MatchFunc
ExAllData:
  $0024   36  sizeof(ExAllData)
  $0000    0  ed_Next
  $0004    4  ed_Name
  $0008    8  ed_Type
  $000c   12  ed_Size
  $0010   16  ed_Prot
  $0014   20  ed_Days
  $0018   24  ed_Mins
  $001c   28  ed_Ticks
  $0020   32  ed_Comment
ExecBase:
  $0268  616  sizeof(ExecBase)
  $0000    0  LibNode
  $0022   34  SoftVer
  $0024   36  LowMemChkSum
! $0026   38  ChkBase
! $002a   42  ColdCapture
! $002e   46  CoolCapture
! $0032   50  WarmCapture
! $0036   54  SysStkUpper
! $003a   58  SysStkLower
! $003e   62  MaxLocMem
```

```
! $0042   66  DebugEntry
! $0046   70  DebugData
! $004a   74  AlertData
! $004e   78  MaxExtMem
  $0052   82  ChkSum
  $0054   84  IntVects[0]
  $0114  276  ThisTask
  $0118  280  IdleCount
  $011c  284  DispCount
  $0120  288  Quantum
  $0122  290  Elapsed
  $0124  292  SysFlags
  $0126  294  IDNestCnt
  $0127  295  TDNestCnt
  $0128  296  AttnFlags
  $012a  298  AttnResched
  $012c  300  ResModules
  $0130  304  TaskTrapCode
  $0134  308  TaskExceptCode
  $0138  312  TaskExitCode
  $013c  316  TaskSigAlloc
  $0140  320  TaskTrapAlloc
! $0142  322  MemList
  $0150  336  ResourceList
! $015e  350  DeviceList
  $016c  364  IntrList
! $017a  378  LibList
  $0188  392  PortList
! $0196  406  TaskReady
  $01a4  420  TaskWait
! $01b2  434  SoftInts[0]
! $0202  514  LastAlert[0]
  $0212  530  VBlankFrequency
  $0213  531  PowerSupplyFrequency
  $0214  532  SemaphoreList
! $0222  546  KickMemPtr
! $0226  550  KickTagPtr
! $022a  554  KickCheckSum
  $022e  558  ex_Pad0
  $0230  560  ex_Reserved0
  $0234  564  ex_RamLibPrivate
  $0238  568  ex_EClockFrequency
  $023c  572  ex_CacheControl
  $0240  576  ex_TaskID
  $0244  580  ex_PuddleSize
  $0248  584  ex_PoolThreshold
  $024c  588  ex_PublicPool
  $0258  600  ex_MMULock
  $025c  604  ex_Reserved[0]
ExpansionBase:
  $0058   88  sizeof(ExpansionBase)
  $0000    0  LibNode
  $0022   34  Flags
  $0023   35  eb_Private01
  $0024   36  eb_Private02
  $0028   40  eb_Private03
  $002c   44  eb_Private04
  $003c   60  eb_Private05
! $004a   74  MountList
ExpansionControl:
  $0010   16  sizeof(ExpansionControl)
  $0000    0  ec_Interrupt
  $0001    1  ec_Z3_HighBase
  $0002    2  ec_BaseAddress
  $0003    3  ec_Shutup
  $0004    4  ec_Reserved14
```

Structure Reference

Page 8

```
  $0005    5  ec_Reserved15
  $0006    6  ec_Reserved16
  $0007    7  ec_Reserved17
  $0008    8  ec_Reserved18
  $0009    9  ec_Reserved19
  $000a   10  ec_Reserved1a
  $000b   11  ec_Reserved1b
  $000c   12  ec_Reserved1c
  $000d   13  ec_Reserved1d
  $000e   14  ec_Reserved1e
  $000f   15  ec_Reserved1f
ExpansionRom:
  $0010   16  sizeof(ExpansionRom)
  $0000    0  er_Type
  $0001    1  er_Product
  $0002    2  er_Flags
  $0003    3  er_Reserved03
  $0004    4  er_Manufacturer
! $0006    6  er_SerialNumber
  $000a   10  er_InitDiagVec
  $000c   12  er_Reserved0c
  $000d   13  er_Reserved0d
  $000e   14  er_Reserved0e
  $000f   15  er_Reserved0f
ExtNewScreen:
  $0024   36  sizeof(ExtNewScreen)
  $0000    0  LeftEdge
  $0002    2  TopEdge
  $0004    4  Width
  $0006    6  Height
  $0008    8  Depth
  $000a   10  DetailPen
  $000b   11  BlockPen
  $000c   12  ViewModes
  $000e   14  Type
  $0010   16  Font
  $0014   20  DefaultTitle
  $0018   24  Gadgets
  $001c   28  CustomBitMap
  $0020   32  Extension
ExtNewWindow:
  $0034   52  sizeof(ExtNewWindow)
  $0000    0  LeftEdge
  $0002    2  TopEdge
  $0004    4  Width
  $0006    6  Height
  $0008    8  DetailPen
  $0009    9  BlockPen
! $000a   10  IDCMPFlags
! $000e   14  Flags
! $0012   18  FirstGadget
! $0016   22  CheckMark
! $001a   26  Title
! $001e   30  Screen
! $0022   34  BitMap
  $0026   38  MinWidth
  $0028   40  MinHeight
  $002a   42  MaxWidth
  $002c   44  MaxHeight
  $002e   46  Type
  $0030   48  Extension
ExtendedNode:
  $0018   24  sizeof(ExtendedNode)
  $0000    0  xln_Succ
  $0004    4  xln_Pred
  $0008    8  xln_Type
```

```
  $0009    9  xln_Pri
! $000a   10  xln_Name
  $000e   14  xln_Subsystem
  $000f   15  xln_Subtype
  $0010   16  xln_Library
  $0014   20  xln_Init
FileHandle:
  $002c   44  sizeof(FileHandle)
  $0000    0  fh_Link
  $0004    4  fh_Port
  $0008    8  fh_Type
  $000c   12  fh_Buf
  $0010   16  fh_Pos
  $0014   20  fh_End
  $0018   24  fh_Funcs
  $001c   28  fh_Func2
  $0020   32  fh_Func3
  $0024   36  fh_Args
  $0028   40  fh_Arg2
FileInfoBlock:
  $0104  260  sizeof(FileInfoBlock)
  $0000    0  fib_DiskKey
  $0004    4  fib_DirEntryType
  $0008    8  fib_FileName[0]
  $0074  116  fib_Protection
  $0078  120  fib_EntryType
  $007c  124  fib_Size
  $0080  128  fib_NumBlocks
  $0084  132  fib_Date
  $0090  144  fib_Comment[0]
  $00e0  224  fib_Reserved[0]
FileLock:
  $0014   20  sizeof(FileLock)
  $0000    0  fl_Link
  $0004    4  fl_Key
  $0008    8  fl_Access
  $000c   12  fl_Task
  $0010   16  fl_Volume
FileRequester:
  $002c   44  sizeof(FileRequester)
  $0000    0  rf_Reserved1
  $0004    4  rf_File
  $0008    8  rf_Dir
  $000c   12  rf_Reserved2[0]
  $0016   22  rf_LeftEdge
  $0018   24  rf_TopEdge
  $001a   26  rf_Width
  $001c   28  rf_Height
  $001e   30  rf_Reserved3
  $0020   32  rf_NumArgs
  $0024   36  rf_ArgList
  $0028   40  rf_UserData
FileSysEntry:
  $003e   62  sizeof(FileSysEntry)
  $0000    0  fse_Node
! $000e   14  fse_DosType
! $0012   18  fse_Version
! $0016   22  fse_PatchFlags
! $001a   26  fse_Type
! $001e   30  fse_Task
! $0022   34  fse_Lock
! $0026   38  fse_Handler
! $002a   42  fse_StackSize
! $002e   46  fse_Priority
! $0032   50  fse_Startup
! $0036   54  fse_SegList
```

Structure Reference Page 9

```
! $003a   58  fse_GlobalVec                     $0010   16  GadgetType
FileSysHeaderBlock:                           ! $0012   18  GadgetRender
  $0100  256  sizeof(FileSysHeaderBlock)      ! $0016   22  SelectRender
  $0000    0  fhb_ID                          ! $001a   26  GadgetText
  $0004    4  fhb_SummedLongs                 ! $001e   30  MutualExclude
  $0008    8  fhb_ChkSum                      ! $0022   34  SpecialInfo
  $000c   12  fhb_HostID                        $0026   38  GadgetID
  $0010   16  fhb_Next                          $0028   40  UserData
  $0014   20  fhb_Flags                       GadgetInfo:
  $0018   24  fhb_Reserved1[0]                  $003a   58  sizeof(GadgetInfo)
  $0020   32  fhb_DosType                       $0000    0  gi_Screen
  $0024   36  fhb_Version                       $0004    4  gi_Window
  $0028   40  fhb_PatchFlags                    $0008    8  gi_Requester
  $002c   44  fhb_Type                          $000c   12  gi_RastPort
  $0030   48  fhb_Task                          $0010   16  gi_Layer
  $0034   52  fhb_Lock                          $0014   20  gi_Domain
  $0038   56  fhb_Handler                       $001c   28  gi_Pens
  $003c   60  fhb_StackSize                     $001c   28  gi_Pens.DetailPen
  $0040   64  fhb_Priority                      $001d   29  gi_Pens.BlockPen
  $0044   68  fhb_Startup                     ! $001e   30  gi_DrInfo
  $0048   72  fhb_SegListBlocks               ! $0022   34  gi_Reserved[0]
  $004c   76  fhb_GlobalVec                   GamePortTrigger:
  $0050   80  fhb_Reserved2[0]                  $0008    8  sizeof(GamePortTrigger)
  $00ac  172  fhb_Reserved3[0]                  $0000    0  gpt_Keys
FileSysResource:                                $0002    2  gpt_Timeout
  $0020   32  sizeof(FileSysResource)           $0004    4  gpt_XDelta
  $0000    0  fsr_Node                          $0006    6  gpt_YDelta
! $000e   14  fsr_Creator                     GelsInfo:
! $0012   18  fsr_FileSysEntries                $0026   38  sizeof(GelsInfo)
FileSysStartupMsg:                              $0000    0  sprRsrvd
  $0010   16  sizeof(FileSysStartupMsg)         $0001    1  Flags
  $0000    0  fssm_Unit                       ! $0002    2  gelHead
  $0004    4  fssm_Device                     ! $0006    6  gelTail
  $0008    8  fssm_Environ                    ! $000a   10  nextLine
  $000c   12  fssm_Flags                      ! $000e   14  lastColor
FontContents:                                 ! $0012   18  collHandler
  $0104  260  sizeof(FontContents)              $0016   22  leftmost
  $0000    0  fc_FileName[0]                    $0018   24  rightmost
  $0100  256  fc_YSize                          $001a   26  topmost
  $0102  258  fc_Style                          $001c   28  bottommost
  $0103  259  fc_Flags                        ! $001e   30  firstBlissObj
FontContentsHeader:                           ! $0022   34  lastBlissObj
  $0004    4  sizeof(FontContentsHeader)      GfxBase:
  $0000    0  fch_FileID                        $01a6  422  sizeof(GfxBase)
  $0002    2  fch_NumEntries                    $0000    0  LibNode
FontRequester:                                ! $0022   34  ActiView
  $0018   24  sizeof(FontRequester)           ! $0026   38  copinit
  $0000    0  fo_Reserved1[0]                 ! $002a   42  cia
  $0008    8  fo_Attr                         ! $002e   46  blitter
  $0010   16  fo_FrontPen                     ! $0032   50  LOFlist
  $0011   17  fo_BackPen                      ! $0036   54  SHFlist
  $0012   18  fo_DrawMode                     ! $003a   58  blthd
  $0014   20  fo_UserData                     ! $003e   62  blttl
FreeList:                                     ! $0042   66  bsblthd
  $0010   16  sizeof(FreeList)                ! $0046   70  bsblttl
  $0000    0  fl_NumFree                      ! $004a   74  vbsrv
! $0002    2  fl_MemList                        $0060   96  timsrv
Gadget:                                       ! $0076  118  bltsrv
  $002c   44  sizeof(Gadget)                    $008c  140  TextFonts
  $0000    0  NextGadget                      ! $009a  154  DefaultFont
  $0004    4  LeftEdge                          $009e  158  Modes
  $0006    6  TopEdge                           $00a0  160  VBlank
  $0008    8  Width                             $00a1  161  Debug
  $000a   10  Height                            $00a2  162  BeamSync
  $000c   12  Flags                             $00a4  164  system_bplcon0
  $000e   14  Activation                        $00a6  166  SpriteReserved
```

Structure Reference Page 10

```
  $00a7  167  bytereserved                      $0000    0  iept_Range.X
  $00a8  168  Flags                             $0002    2  iept_Range.Y
  $00aa  170  BlitLock                          $0004    4  iept_Value
  $00ac  172  BlitNest                          $0004    4  iept_Value.X
! $00ae  174  BlitWaitQ                         $0006    6  iept_Value.Y
  $00bc  188  BlitOwner                         $0008    8  iept_Pressure
  $00c0  192  TOF_WaitQ                       IFFHandle:
  $00ce  206  DisplayFlags                      $000c   12  sizeof(IFFHandle)
  $00d0  208  SimpleSprites                     $0000    0  iff_Stream
  $00d4  212  MaxDisplayRow                     $0004    4  iff_Flags
  $00d6  214  MaxDisplayColumn                  $0008    8  iff_Depth
  $00d8  216  NormalDisplayRows               IFFStreamCmd:
  $00da  218  NormalDisplayColumns              $000c   12  sizeof(IFFStreamCmd)
  $00dc  220  NormalDPMX                        $0000    0  sc_Command
  $00de  222  NormalDPMY                        $0004    4  sc_Buf
  $00e0  224  LastChanceMemory                  $0008    8  sc_NBytes
  $00e4  228  LCMptr                          IOAudio:
  $00e8  232  MicrosPerLine                     $0044   68  sizeof(IOAudio)
  $00ea  234  MinDisplayColumn                  $0000    0  ioa_Request
  $00ec  236  ChipRevBits0                      $0020   32  ioa_AllocKey
! $00ed  237  crb_reserved[0]                 ! $0022   34  ioa_Data
  $00f2  242  monitor_id                      ! $0026   38  ioa_Length
  $00f4  244  hedley[0]                         $002a   42  ioa_Period
  $0114  276  hedley_sprites[0]                 $002c   44  ioa_Volume
  $0134  308  hedley_sprites1[0]                $002e   46  ioa_Cycles
  $0154  340  hedley_count                      $0030   48  ioa_WriteMsg
  $0156  342  hedley_flags                    IOClipReq:
  $0158  344  hedley_tmp                        $0034   52  sizeof(IOClipReq)
! $015a  346  hash_table                        $0000    0  io_Message
  $015e  350  current_tot_rows                  $0014   20  io_Device
  $0160  352  current_tot_cclks                 $0018   24  io_Unit
  $0162  354  hedley_hint                       $001c   28  io_Command
  $0163  355  hedley_hint2                      $001e   30  io_Flags
  $0164  356  nreserved[0]                      $001f   31  io_Error
  $0174  372  a2024_sync_raster                 $0020   32  io_Actual
  $0178  376  control_delta_pal                 $0024   36  io_Length
  $017a  378  control_delta_ntsc                $0028   40  io_Data
  $017c  380  current_monitor                   $002c   44  io_Offset
  $0180  384  MonitorList                       $0030   48  io_ClipID
! $018e  398  default_monitor                 IODRPReq:
! $0192  402  MonitorListSemaphore              $003e   62  sizeof(IODRPReq)
! $0196  406  DisplayInfoDataBase               $0000    0  io_Message
! $019a  410  ActiViewCprSemaphore              $0014   20  io_Device
! $019e  414  UtilityBase                       $0018   24  io_Unit
! $01a2  418  ExecBase                          $001c   28  io_Command
Hook:                                           $001e   30  io_Flags
  $0014   20  sizeof(Hook)                      $001f   31  io_Error
  $0000    0  h_MinNode                         $0020   32  io_RastPort
  $0008    8  h_Entry                           $0024   36  io_ColorMap
  $000c   12  h_SubEntry                        $0028   40  io_Modes
  $0010   16  h_Data                            $002c   44  io_SrcX
IBox:                                           $002e   46  io_SrcY
  $0008    8  sizeof(IBox)                      $0030   48  io_SrcWidth
  $0000    0  Left                              $0032   50  io_SrcHeight
  $0002    2  Top                               $0034   52  io_DestCols
  $0004    4  Width                             $0038   56  io_DestRows
  $0006    6  Height                            $003c   60  io_Special
IEPointerPixel:                               IOExtPar:
  $0008    8  sizeof(IEPointerPixel)            $003e   62  sizeof(IOExtPar)
  $0000    0  iepp_Screen                       $0000    0  IOPar
  $0004    4  iepp_Position                     $0030   48  io_PExtFlags
  $0004    4  iepp_Position.X                   $0034   52  io_Status
  $0006    6  iepp_Position.Y                   $0035   53  io_ParFlags
IEPointerTablet:                              ! $0036   54  io_PTermArray
  $000a   10  sizeof(IEPointerTablet)         IOExtSer:
  $0000    0  iept_Range                        $0052   82  sizeof(IOExtSer)
```

Structure Reference Page 11

```
  $0000    0  IOSer
  $0030   48  io_CtlChar
  $0034   52  io_RBufLen
  $0038   56  io_ExtFlags
  $003c   60  io_Baud
  $0040   64  io_BrkTime
  $0044   68  io_TermArray
  $004c   76  io_ReadLen
  $004d   77  io_WriteLen
  $004e   78  io_StopBits
  $004f   79  io_SerFlags
  $0050   80  io_Status
IOExtTD:
  $0038   56  sizeof(IOExtTD)
  $0000    0  iotd_Req
  $0030   48  iotd_Count
  $0034   52  iotd_SecLabel
IOPArray:
  $0008    8  sizeof(IOPArray)
  $0000    0  PTermArray0
  $0004    4  PTermArray1
IOPrtCmdReq:
  $0026   38  sizeof(IOPrtCmdReq)
  $0000    0  io_Message
  $0014   20  io_Device
  $0018   24  io_Unit
  $001c   28  io_Command
  $001e   30  io_Flags
  $001f   31  io_Error
  $0020   32  io_PrtCommand
  $0022   34  io_Parm0
  $0023   35  io_Parm1
  $0024   36  io_Parm2
  $0025   37  io_Parm3
IORequest:
  $0020   32  sizeof(IORequest)
  $0000    0  io_Message
  $0014   20  io_Device
  $0018   24  io_Unit
  $001c   28  io_Command
  $001e   30  io_Flags
  $001f   31  io_Error
IOStdReq:
  $0030   48  sizeof(IOStdReq)
  $0000    0  io_Message
  $0014   20  io_Device
  $0018   24  io_Unit
  $001c   28  io_Command
  $001e   30  io_Flags
  $001f   31  io_Error
  $0020   32  io_Actual
  $0024   36  io_Length
  $0028   40  io_Data
  $002c   44  io_Offset
IOTArray:
  $0008    8  sizeof(IOTArray)
  $0000    0  TermArray0
  $0004    4  TermArray1
Image:
  $0014   20  sizeof(Image)
  $0000    0  LeftEdge
  $0002    2  TopEdge
  $0004    4  Width
  $0006    6  Height
  $0008    8  Depth
! $000a   10  ImageData
```

```
  $000e   14  PlanePick
  $000f   15  PlaneOnOff
  $0010   16  NextImage
InfoData:
  $0024   36  sizeof(InfoData)
  $0000    0  id_NumSoftErrors
  $0004    4  id_UnitNumber
  $0008    8  id_DiskState
  $000c   12  id_NumBlocks
  $0010   16  id_NumBlocksUsed
  $0014   20  id_BytesPerBlock
  $0018   24  id_DiskType
  $001c   28  id_VolumeNode
  $0020   32  id_InUse
InputEvent:
  $0016   22  sizeof(InputEvent)
  $0000    0  ie_NextEvent
  $0004    4  ie_Class
  $0005    5  ie_SubClass
  $0006    6  ie_Code
  $0008    8  ie_Qualifier
  $000a   10  ie_position
  $000a   10  ie_position.ie_xy
  $000a   10  ie_position.ie_xy.ie_x
  $000c   12  ie_position.ie_xy.ie_y
  $000a   10  ie_position.ie_addr
  $000a   10  ie_position.ie_dead
  $000a   10
     ie_position.ie_dead.ie_prev1DownCode
  $000b   11
     ie_position.ie_dead.ie_prev1DownQual
  $000c   12
     ie_position.ie_dead.ie_prev2DownCode
  $000d   13
     ie_position.ie_dead.ie_prev2DownQual
! $000e   14  ie_TimeStamp
InputXpression:
  $000c   12  sizeof(InputXpression)
  $0000    0  ix_Version
  $0001    1  ix_Class
  $0002    2  ix_Code
  $0004    4  ix_CodeMask
  $0006    6  ix_Qualifier
  $0008    8  ix_QualMask
  $000a   10  ix_QualSame
IntVector:
  $000c   12  sizeof(IntVector)
  $0000    0  iv_Data
  $0004    4  iv_Code
  $0008    8  iv_Node
Interrupt:
  $0016   22  sizeof(Interrupt)
  $0000    0  is_Node
! $000e   14  is_Data
! $0012   18  is_Code
IntuiMessage:
  $0034   52  sizeof(IntuiMessage)
  $0000    0  ExecMessage
  $0014   20  Class
  $0018   24  Code
  $001a   26  Qualifier
  $001c   28  IAddress
  $0020   32  MouseX
  $0022   34  MouseY
  $0024   36  Seconds
  $0028   40  Micros
```

Structure Reference Page 12

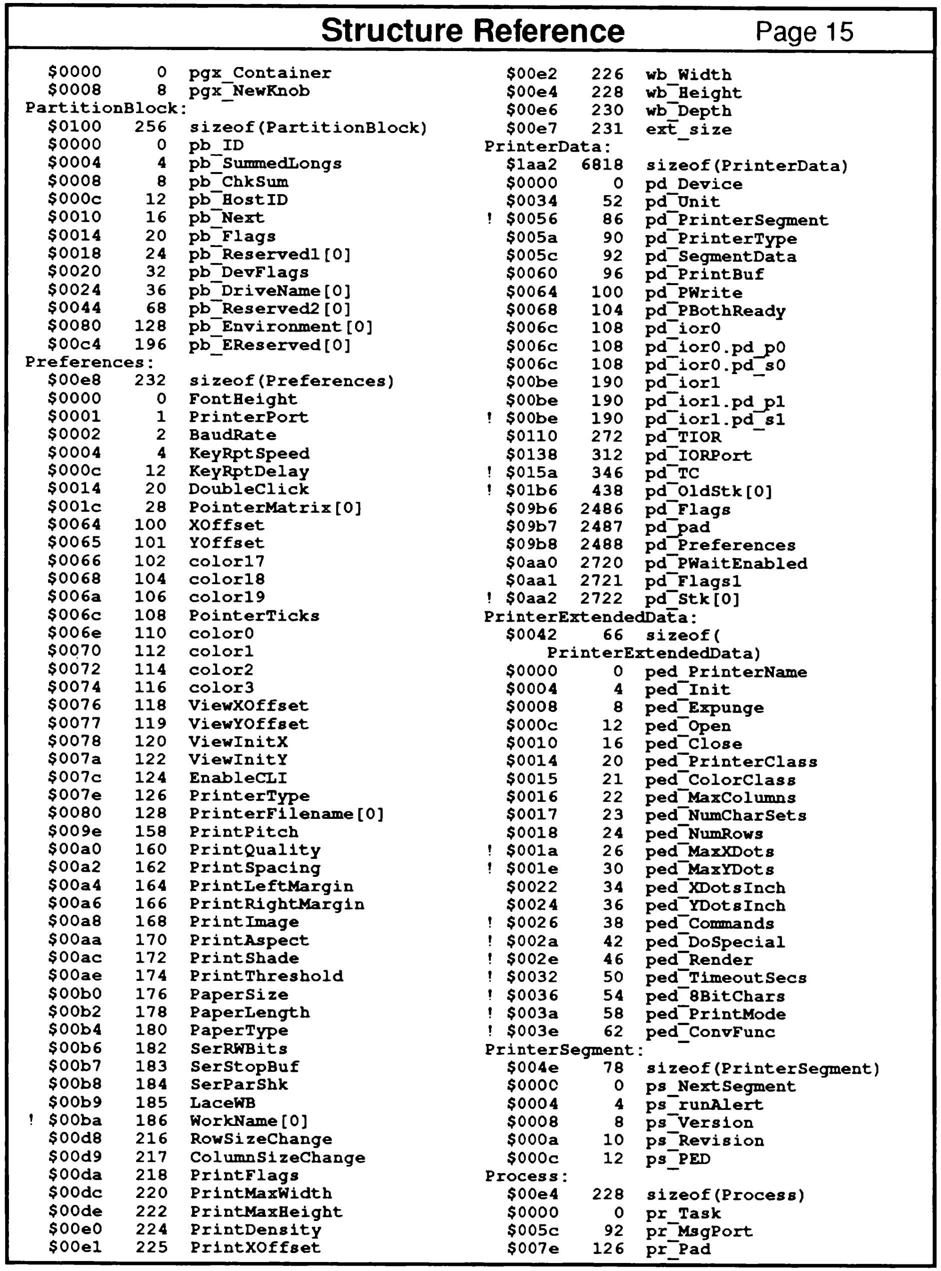
```
  $002c   44  IDCMPWindow
  $0030   48  SpecialLink
IntuiText: .
  $0014   20  sizeof(IntuiText)
  $0000    0  FrontPen
  $0001    1  BackPen
  $0002    2  DrawMode
  $0004    4  LeftEdge
  $0006    6  TopEdge
  $0008    8  ITextFont
  $000c   12  IText
  $0010   16  NextText
IntuitionBase:
  $0050   80  sizeof(IntuitionBase)
  $0000    0  LibNode
! $0022   34  ViewLord
  $0034   52  ActiveWindow
  $0038   56  ActiveScreen
  $003c   60  FirstScreen
  $0040   64  Flags
  $0044   68  MouseY
  $0046   70  MouseX
  $0048   72  Seconds
  $004c   76  Micros
IoBuff:
  $0100  256  sizeof(IoBuff)
  $0000    0  iobNode
  $0020   32  iobRpt
  $0024   36  iobRct
  $0028   40  iobDFH
  $002c   44  iobLock
  $0030   48  iobBct
  $0034   52  iobArea[0]
Isrvstr:
  $001e   30  sizeof(Isrvstr)
  $0000    0  is_Node
! $000e   14  Iptr
! $0012   18  code
! $0016   22  ccode
! $001a   26  Carg
KeyMap:
  $0020   32  sizeof(KeyMap)
  $0000    0  km_LoKeyMapTypes
  $0004    4  km_LoKeyMap
  $0008    8  km_LoCapsable
  $000c   12  km_LoRepeatable
  $0010   16  km_HiKeyMapTypes
  $0014   20  km_HiKeyMap
  $0018   24  km_HiCapsable
  $001c   28  km_HiRepeatable
KeyMapNode:
  $002e   46  sizeof(KeyMapNode)
  $0000    0  kn_Node
! $000e   14  kn_KeyMap
KeyMapResource:
  $001c   28  sizeof(KeyMapResource)
  $0000    0  kr_Node
! $000e   14  kr_List
Layer:
  $00a0  160  sizeof(Layer)
  $0000    0  front
  $0004    4  back
  $0008    8  ClipRect
  $000c   12  rp
  $0010   16  bounds
  $0018   24  reserved[0]
```

```
  $001c   28  priority
  $001e   30  Flags
  $0020   32  SuperBitMap
  $0024   36  SuperClipRect
  $0028   40  Window
  $002c   44  Scroll_X
  $002e   46  Scroll_Y
  $0030   48  cr
  $0034   52  cr2
  $0038   56  crnew
  $003c   60  SuperSaveClipRects
  $0040   64  _cliprects
  $0044   68  LayerInfo
  $0048   72  Lock
! $0076  118  BackFill
! $007a  122  reserved1
! $007e  126  ClipRegion
! $0082  130  saveClipRects
  $0086  134  Width
  $0088  136  Height
! $008a  138  reserved2[0]
  $009c  156  DamageList
Layer_Info:
  $0066  102  sizeof(Layer_Info)
  $0000    0  top_layer
  $0004    4  check_lp
  $0008    8  obs
  $000c   12  FreeClipRects
  $0018   24  Lock
! $0046   70  gs_Head
  $0054   84  longreserved
  $0058   88  Flags
  $005a   90  fatten_count
  $005b   91  LockLayersCount
  $005c   92  LayerInfo_extra_size
! $005e   94  blitbuff
! $0062   98  LayerInfo_extra
Library:
  $0022   34  sizeof(Library)
  $0000    0  lib_Node
  $000e   14  lib_Flags
  $000f   15  lib_pad
  $0010   16  lib_NegSize
  $0012   18  lib_PosSize
  $0014   20  lib_Version
  $0016   22  lib_Revision
  $0018   24  lib_IdString
  $001c   28  lib_Sum
  $0020   32  lib_OpenCnt
List:
  $000e   14  sizeof(List)
  $0000    0  lh_Head
  $0004    4  lh_Tail
  $0008    8  lh_TailPred
  $000c   12  lh_Type
  $000d   13  l_pad
LoadSegBlock:
  $0200  512  sizeof(LoadSegBlock)
  $0000    0  lsb_ID
  $0004    4  lsb_SummedLongs
  $0008    8  lsb_ChkSum
  $000c   12  lsb_HostID
  $0010   16  lsb_Next
  $0014   20  lsb_LoadData[0]
LocalContextItem:
  $0014   20  sizeof(LocalContextItem)
  $0000     0  lci_Node
  $0008     8  lci_ID
  $000c    12  lci_Type
  $0010    16  lci_Ident
LocalVar:
  $0018    24  sizeof(LocalVar)
  $0000     0  lv_Node
  $000e    14  lv_Flags
  $0010    16  lv_Value
  $0014    20  lv_Len
MathIEEEBase:
  $003c    60  sizeof(MathIEEEBase)
  $0000     0  MathIEEEBase_LibNode
! $0022    34  MathIEEEBase_reserved[0]
  $0034    52  MathIEEEBase_TaskOpenLib
  $0038    56  MathIEEEBase_TaskCloseLib
MathIEEEResource:
  $002c    44  sizeof(MathIEEEResource)
  $0000     0  MathIEEEResource_Node
  $000e    14  MathIEEEResource_Flags
  $0010    16  MathIEEEResource_BaseAddr
  $0014    20
       MathIEEEResource_DblBasInit
  $0018    24
       MathIEEEResource_DblTransInit
  $001c    28
       MathIEEEResource_SglBasInit
  $0020    32
       MathIEEEResource_SglTransInit
  $0024    36
       MathIEEEResource_ExtBasInit
  $0028    40
       MathIEEEResource_ExtTransInit
MemChunk:
  $0008     8  sizeof(MemChunk)
  $0000     0  mc_Next
  $0004     4  mc_Bytes
MemEntry:
  $0008     8  sizeof(MemEntry)
  $0000     0  me_Un
  $0000     0  me_Un.meu_Reqs
  $0000     0  me_Un.meu_Addr
  $0004     4  me_Length
MemHeader:
  $0020    32  sizeof(MemHeader)
  $0000     0  mh_Node
  $000e    14  mh_Attributes
  $0010    16  mh_First
  $0014    20  mh_Lower
  $0018    24  mh_Upper
  $001c    28  mh_Free
MemList:
  $0018    24  sizeof(MemList)
  $0000     0  ml_Node
  $000e    14  ml_NumEntries
  $0010    16  ml_ME[0]
Menu:
  $001e    30  sizeof(Menu)
  $0000     0  NextMenu
  $0004     4  LeftEdge
  $0006     6  TopEdge
  $0008     8  Width
  $000a    10  Height
  $000c    12  Flags
! $000e    14  MenuName
! $0012    18  FirstItem
```

```
  $0016    22  JazzX
  $0018    24  JazzY
  $001a    26  BeatX
  $001c    28  BeatY
MenuItem:
  $0022    34  sizeof(MenuItem)
  $0000     0  NextItem
  $0004     4  LeftEdge
  $0006     6  TopEdge
  $0008     8  Width
  $000a    10  Height
  $000c    12  Flags
! $000e    14  MutualExclude
! $0012    18  ItemFill
! $0016    22  SelectFill
  $001a    26  Command
  $001c    28  SubItem
  $0020    32  NextSelect
Message:
  $0014    20  sizeof(Message)
  $0000     0  mn_Node
! $000e    14  mn_ReplyPort
  $0012    18  mn_Length
MinList:
  $000c    12  sizeof(MinList)
  $0000     0  mlh_Head
  $0004     4  mlh_Tail
  $0008     8  mlh_TailPred
MinNode:
  $0008     8  sizeof(MinNode)
  $0000     0  mln_Succ
  $0004     4  mln_Pred
MonitorInfo:
  $0058    88  sizeof(MonitorInfo)
  $0000     0  Header
  $0010    16  Mspc
  $0014    20  ViewPosition
  $0018    24  ViewResolution
  $001c    28  ViewPositionRange
  $0024    36  TotalRows
  $0026    38  TotalColorClocks
  $0028    40  MinRow
  $002a    42  Compatibility
  $002c    44  pad[0]
  $0050    80  reserved[0]
MonitorSpec:
  $009c   156  sizeof(MonitorSpec)
  $0000     0  ms_Node
  $0018    24  ms_Flags
! $001a    26  ratioh
! $001e    30  ratiov
  $0022    34  total_rows
  $0024    36  total_colorclocks
  $0026    38  DeniseMaxDisplayColumn
  $0028    40  BeamCon0
  $002a    42  min_row
  $002c    44  ms_Special
  $0030    48  ms_OpenCount
! $0032    50  ms_transform
! $0036    54  ms_translate
! $003a    58  ms_scale
  $003e    62  ms_xoffset
  $0040    64  ms_yoffset
! $0042    66  ms_LegalView
! $004a    74  ms_maxoscan
! $004e    78  ms_videoscan
```

```
  $0052    82  DeniseMinDisplayColumn
  $0054    84  DisplayCompatible
  $0058    88  DisplayInfoDataBase
! $0066   102
       DisplayInfoDataBaseSemaphore
  $0094   148  ms_reserved00
  $0098   152  ms_reserved01
MsgPort:
  $0022    34  sizeof(MsgPort)
  $0000     0  mp_Node
  $000e    14  mp_Flags
  $000f    15  mp_SigBit
  $0010    16  mp_SigTask
  $0014    20  mp_MsgList
NameInfo:
  $0038    56  sizeof(NameInfo)
  $0000     0  Header
  $0010    16  Name[0]
  $0030    48  reserved[0]
NewBroker:
  $001a    26  sizeof(NewBroker)
  $0000     0  nb_Version
! $0002     2  nb_Name
! $0006     6  nb_Title
! $000a    10  nb_Descr
  $000e    14  nb_Unique
  $0010    16  nb_Flags
  $0012    18  nb_Pri
  $0014    20  nb_Port
  $0018    24  nb_ReservedChannel
NewGadget:
  $001e    30  sizeof(NewGadget)
  $0000     0  ng_LeftEdge
  $0002     2  ng_TopEdge
  $0004     4  ng_Width
  $0006     6  ng_Height
  $0008     8  ng_GadgetText
  $000c    12  ng_TextAttr
  $0010    16  ng_GadgetID
! $0012    18  ng_Flags
! $0016    22  ng_VisualInfo
! $001a    26  ng_UserData
NewMenu:
  $0014    20  sizeof(NewMenu)
  $0000     0  nm_Type
! $0002     2  nm_Label
! $0006     6  nm_CommKey
  $000a    10  nm_Flags
  $000c    12  nm_MutualExclude
  $0010    16  nm_UserData
NewScreen:
  $0020    32  sizeof(NewScreen)
  $0000     0  LeftEdge
  $0002     2  TopEdge
  $0004     4  Width
  $0006     6  Height
  $0008     8  Depth
  $000a    10  DetailPen
  $000b    11  BlockPen
  $000c    12  ViewModes
  $000e    14  Type
  $0010    16  Font
  $0014    20  DefaultTitle
  $0018    24  Gadgets
  $001c    28  CustomBitMap
NewWindow:
```

```
  $0030    48  sizeof(NewWindow)
  $0000     0  LeftEdge
  $0002     2  TopEdge
  $0004     4  Width
  $0006     6  Height
  $0008     8  DetailPen
  $0009     9  BlockPen
! $000a    10  IDCMPFlags
! $000e    14  Flags
! $0012    18  FirstGadget
! $0016    22  CheckMark
! $001a    26  Title
! $001e    30  Screen
! $0022    34  BitMap
  $0026    38  MinWidth
  $0028    40  MinHeight
  $002a    42  MaxWidth
  $002c    44  MaxHeight
  $002e    46  Type
NexxStr:
  $0010    16  sizeof(NexxStr)
  $0000     0  ns_Ivalue
  $0004     4  ns_Length
  $0006     6  ns_Flags
  $0007     7  ns_Hash
  $0008     8  ns_Buff[0]
Node:
  $000e    14  sizeof(Node)
  $0000     0  ln_Succ
  $0004     4  ln_Pred
  $0008     8  ln_Type
  $0009     9  ln_Pri
! $000a    10  ln_Name
NotifyMessage:
  $0026    38  sizeof(NotifyMessage)
  $0000     0  nm_ExecMessage
  $0014    20  nm_Class
  $0018    24  nm_Code
! $001a    26  nm_NReq
! $001e    30  nm_DoNotTouch
! $0022    34  nm_DoNotTouch2
NotifyRequest:
  $0030    48  sizeof(NotifyRequest)
  $0000     0  nr_Name
  $0004     4  nr_FullName
  $0008     8  nr_UserData
  $000c    12  nr_Flags
  $0010    16  nr_stuff
  $0010    16  nr_stuff.nr_Msg
  $0010    16  nr_stuff.nr_Msg.nr_Port
  $0010    16  nr_stuff.nr_Signal
  $0010    16  nr_stuff.nr_Signal.nr_Task
  $0014    20
       nr_stuff.nr_Signal.nr_SignalNum
  $0015    21
       nr_stuff.nr_Signal.nr_pad[0]
  $0018    24  nr_Reserved[0]
  $0028    40  nr_MsgCount
  $002c    44  nr_Handler
OldDrawerData:
  $0038    56  sizeof(OldDrawerData)
  $0000     0  dd_NewWindow
  $0030    48  dd_CurrentX
  $0034    52  dd_CurrentY
PGX:
  $0010    16  sizeof(PGX)
```

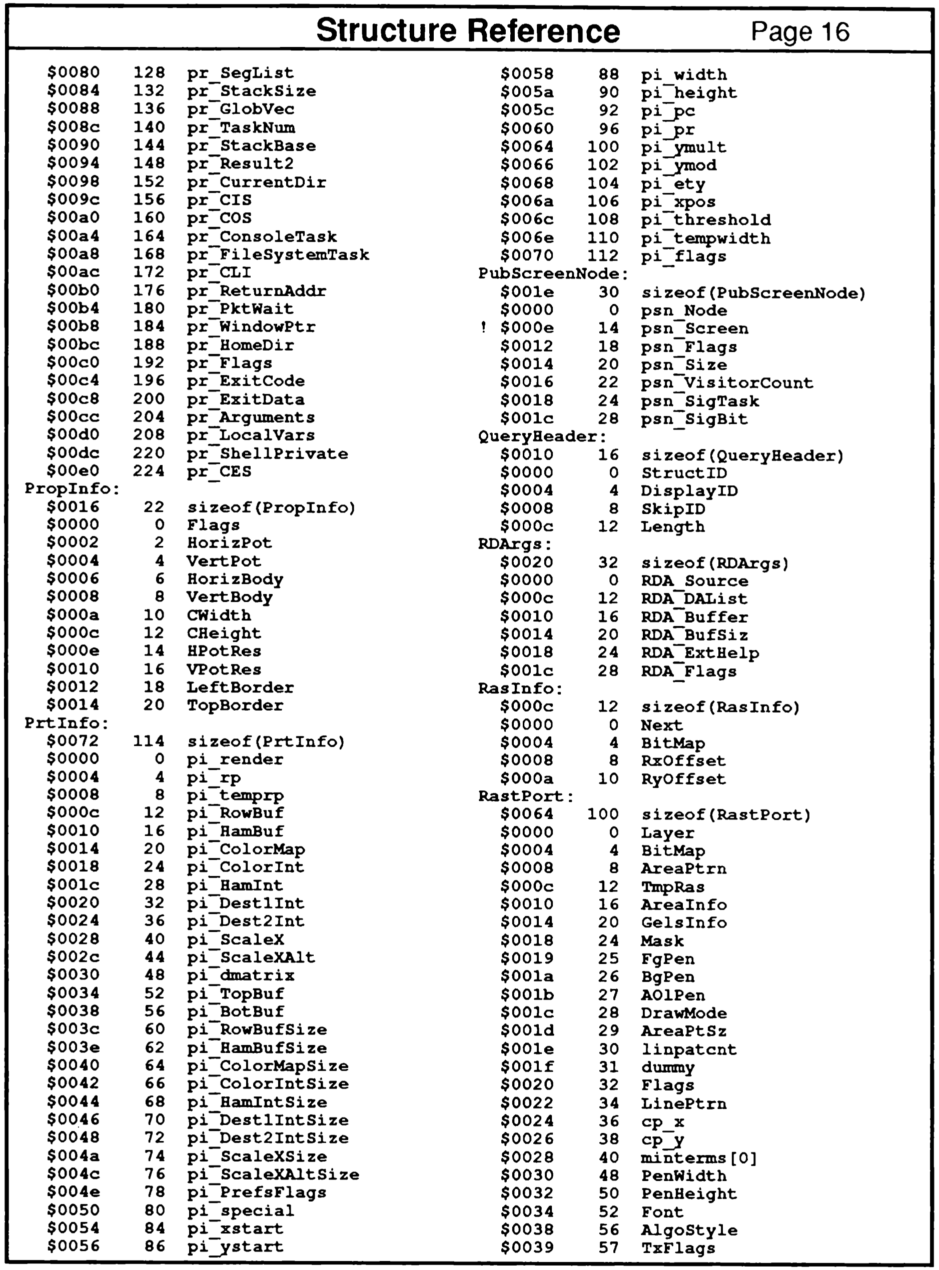


Structure Reference

Page 15

```
  $0000     0  pgx_Container
  $0008     8  pgx_NewKnob
PartitionBlock:
  $0100   256  sizeof(PartitionBlock)
  $0000     0  pb_ID
  $0004     4  pb_SummedLongs
  $0008     8  pb_ChkSum
  $000c    12  pb_HostID
  $0010    16  pb_Next
  $0014    20  pb_Flags
  $0018    24  pb_Reserved1[0]
  $0020    32  pb_DevFlags
  $0024    36  pb_DriveName[0]
  $0044    68  pb_Reserved2[0]
  $0080   128  pb_Environment[0]
  $00c4   196  pb_EReserved[0]
Preferences:
  $00e8   232  sizeof(Preferences)
  $0000     0  FontHeight
  $0001     1  PrinterPort
  $0002     2  BaudRate
  $0004     4  KeyRptSpeed
  $000c    12  KeyRptDelay
  $0014    20  DoubleClick
  $001c    28  PointerMatrix[0]
  $0064   100  XOffset
  $0065   101  YOffset
  $0066   102  color17
  $0068   104  color18
  $006a   106  color19
  $006c   108  PointerTicks
  $006e   110  color0
  $0070   112  color1
  $0072   114  color2
  $0074   116  color3
  $0076   118  ViewXOffset
  $0077   119  ViewYOffset
  $0078   120  ViewInitX
  $007a   122  ViewInitY
  $007c   124  EnableCLI
  $007e   126  PrinterType
  $0080   128  PrinterFilename[0]
  $009e   158  PrintPitch
  $00a0   160  PrintQuality
  $00a2   162  PrintSpacing
  $00a4   164  PrintLeftMargin
  $00a6   166  PrintRightMargin
  $00a8   168  PrintImage
  $00aa   170  PrintAspect
  $00ac   172  PrintShade
  $00ae   174  PrintThreshold
  $00b0   176  PaperSize
  $00b2   178  PaperLength
  $00b4   180  PaperType
  $00b6   182  SerRWBits
  $00b7   183  SerStopBuf
  $00b8   184  SerParShk
  $00b9   185  LaceWB
! $00ba   186  WorkName[0]
  $00d8   216  RowSizeChange
  $00d9   217  ColumnSizeChange
  $00da   218  PrintFlags
  $00dc   220  PrintMaxWidth
  $00de   222  PrintMaxHeight
  $00e0   224  PrintDensity
  $00e1   225  PrintXOffset
```

```
  $00e2   226  wb_Width
  $00e4   228  wb_Height
  $00e6   230  wb_Depth
  $00e7   231  ext_size
PrinterData:
  $1aa2  6818  sizeof(PrinterData)
  $0000     0  pd_Device
  $0034    52  pd_Unit
! $0056    86  pd_PrinterSegment
  $005a    90  pd_PrinterType
  $005c    92  pd_SegmentData
  $0060    96  pd_PrintBuf
  $0064   100  pd_PWrite
  $0068   104  pd_PBothReady
  $006c   108  pd_ior0
  $006c   108  pd_ior0.pd_p0
  $006c   108  pd_ior0.pd_s0
  $00be   190  pd_ior1
  $00be   190  pd_ior1.pd_p1
! $00be   190  pd_ior1.pd_s1
  $0110   272  pd_TIOR
  $0138   312  pd_IORPort
! $015a   346  pd_TC
! $01b6   438  pd_OldStk[0]
  $09b6  2486  pd_Flags
  $09b7  2487  pd_pad
  $09b8  2488  pd_Preferences
  $0aa0  2720  pd_PWaitEnabled
  $0aa1  2721  pd_Flags1
! $0aa2  2722  pd_Stk[0]
PrinterExtendedData:
  $0042    66  sizeof(
      PrinterExtendedData)
  $0000     0  ped_PrinterName
  $0004     4  ped_Init
  $0008     8  ped_Expunge
  $000c    12  ped_Open
  $0010    16  ped_Close
  $0014    20  ped_PrinterClass
  $0015    21  ped_ColorClass
  $0016    22  ped_MaxColumns
  $0017    23  ped_NumCharSets
  $0018    24  ped_NumRows
! $001a    26  ped_MaxXDots
! $001e    30  ped_MaxYDots
  $0022    34  ped_XDotsInch
  $0024    36  ped_YDotsInch
! $0026    38  ped_Commands
! $002a    42  ped_DoSpecial
! $002e    46  ped_Render
! $0032    50  ped_TimeoutSecs
! $0036    54  ped_8BitChars
! $003a    58  ped_PrintMode
! $003e    62  ped_ConvFunc
PrinterSegment:
  $004e    78  sizeof(PrinterSegment)
  $000C     0  ps_NextSegment
  $0004     4  ps_runAlert
  $0008     8  ps_Version
  $000a    10  ps_Revision
  $000c    12  ps_PED
Process:
  $00e4   228  sizeof(Process)
  $0000     0  pr_Task
  $005c    92  pr_MsgPort
  $007e   126  pr_Pad
```



Structure Reference

Page 16

```
  $0080   128  pr_SegList
  $0084   132  pr_StackSize
  $0088   136  pr_GlobVec
  $008c   140  pr_TaskNum
  $0090   144  pr_StackBase
  $0094   148  pr_Result2
  $0098   152  pr_CurrentDir
  $009c   156  pr_CIS
  $00a0   160  pr_COS
  $00a4   164  pr_ConsoleTask
  $00a8   168  pr_FileSystemTask
  $00ac   172  pr_CLI
  $00b0   176  pr_ReturnAddr
  $00b4   180  pr_PktWait
  $00b8   184  pr_WindowPtr
  $00bc   188  pr_HomeDir
  $00c0   192  pr_Flags
  $00c4   196  pr_ExitCode
  $00c8   200  pr_ExitData
  $00cc   204  pr_Arguments
  $00d0   208  pr_LocalVars
  $00dc   220  pr_ShellPrivate
  $00e0   224  pr_CES
PropInfo:
  $0016    22  sizeof(PropInfo)
  $0000     0  Flags
  $0002     2  HorizPot
  $0004     4  VertPot
  $0006     6  HorizBody
  $0008     8  VertBody
  $000a    10  CWidth
  $000c    12  CHeight
  $000e    14  HPotRes
  $0010    16  VPotRes
  $0012    18  LeftBorder
  $0014    20  TopBorder
PrtInfo:
  $0072   114  sizeof(PrtInfo)
  $0000     0  pi_render
  $0004     4  pi_rp
  $0008     8  pi_temprp
  $000c    12  pi_RowBuf
  $0010    16  pi_HamBuf
  $0014    20  pi_ColorMap
  $0018    24  pi_ColorInt
  $001c    28  pi_HamInt
  $0020    32  pi_Dest1Int
  $0024    36  pi_Dest2Int
  $0028    40  pi_ScaleX
  $002c    44  pi_ScaleXAlt
  $0030    48  pi_dmatrix
  $0034    52  pi_TopBuf
  $0038    56  pi_BotBuf
  $003c    60  pi_RowBufSize
  $003e    62  pi_HamBufSize
  $0040    64  pi_ColorMapSize
  $0042    66  pi_ColorIntSize
  $0044    68  pi_HamIntSize
  $0046    70  pi_Dest1IntSize
  $0048    72  pi_Dest2IntSize
  $004a    74  pi_ScaleXSize
  $004c    76  pi_ScaleXAltSize
  $004e    78  pi_PrefsFlags
  $0050    80  pi_special
  $0054    84  pi_xstart
  $0056    86  pi_ystart
```

```
  $0058    88  pi_width
  $005a    90  pi_height
  $005c    92  pi_pc
  $0060    96  pi_pr
  $0064   100  pi_ymult
  $0066   102  pi_ymod
  $0068   104  pi_ety
  $006a   106  pi_xpos
  $006c   108  pi_threshold
  $006e   110  pi_tempwidth
  $0070   112  pi_flags
PubScreenNode:
  $001e    30  sizeof(PubScreenNode)
  $0000     0  psn_Node
! $000e    14  psn_Screen
  $0012    18  psn_Flags
  $0014    20  psn_Size
  $0016    22  psn_VisitorCount
  $0018    24  psn_SigTask
  $001c    28  psn_SigBit
QueryHeader:
  $0010    16  sizeof(QueryHeader)
  $0000     0  StructID
  $0004     4  DisplayID
  $0008     8  SkipID
  $000c    12  Length
RDArgs:
  $0020    32  sizeof(RDArgs)
  $0000     0  RDA_Source
  $000c    12  RDA_DAList
  $0010    16  RDA_Buffer
  $0014    20  RDA_BufSiz
  $0018    24  RDA_ExtHelp
  $001c    28  RDA_Flags
RasInfo:
  $000c    12  sizeof(RasInfo)
  $0000     0  Next
  $0004     4  BitMap
  $0008     8  RxOffset
  $000a    10  RyOffset
RastPort:
  $0064   100  sizeof(RastPort)
  $0000     0  Layer
  $0004     4  BitMap
  $0008     8  AreaPtrn
  $000c    12  TmpRas
  $0010    16  AreaInfo
  $0014    20  GelsInfo
  $0018    24  Mask
  $0019    25  FgPen
  $001a    26  BgPen
  $001b    27  AOlPen
  $001c    28  DrawMode
  $001d    29  AreaPtSz
  $001e    30  linpatcnt
  $001f    31  dummy
  $0020    32  Flags
  $0022    34  LinePtrn
  $0024    36  cp_x
  $0026    38  cp_y
  $0028    40  minterms[0]
  $0030    48  PenWidth
  $0032    50  PenHeight
  $0034    52  Font
  $0038    56  AlgoStyle
  $0039    57  TxFlags
```

Structure Reference

```
  $003a   58  TxHeight
  $003c   60  TxWidth
  $003e   62  TxBaseline
  $0040   64  TxSpacing
! $0042   66  RP_User
! $0046   70  longreserved[0]
! $004e   78  wordreserved[0]
  $005c   92  reserved[0]
RecordLock:
  $0010   16  sizeof(RecordLock)
  $0000    0  rec_FH
  $0004    4  rec_Offset
  $0008    8  rec_Length
  $000c   12  rec_Mode
Rect32:
  $0010   16  sizeof(Rect32)
  $0000    0  MinX
  $0004    4  MinY
  $0008    8  MaxX
  $000c   12  MaxY
Rectangle:
  $0008    8  sizeof(Rectangle)
  $0000    0  MinX
  $0002    2  MinY
  $0004    4  MaxX
  $0006    6  MaxY
Region:
  $000c   12  sizeof(Region)
  $0000    0  bounds
  $0008    8  RegionRectangle
RegionRectangle:
  $0010   16  sizeof(RegionRectangle)
  $0000    0  Next
  $0004    4  Prev
  $0008    8  bounds
Remember:
  $000c   12  sizeof(Remember)
  $0000    0  NextRemember
  $0004    4  RememberSize
  $0008    8  Memory
Requester:
  $0070  112  sizeof(Requester)
  $0000    0  OlderRequest
  $0004    4  LeftEdge
  $0006    6  TopEdge
  $0008    8  Width
  $000a   10  Height
  $000c   12  RelLeft
  $000e   14  RelTop
  $0010   16  ReqGadget
  $0014   20  ReqBorder
  $0018   24  ReqText
  $001c   28  Flags
  $001e   30  BackFill
  $0020   32  ReqLayer
  $0024   36  ReqPad1[0]
  $0044   68  ImageBMap
  $0048   72  RWindow
  $004c   76  ReqImage
  $0050   80  ReqPad2[0]
Resident:
  $001a   26  sizeof(Resident)
  $0000    0  rt_MatchWord
! $0002    2  rt_MatchTag
! $0006    6  rt_EndSkip
  $000a   10  rt_Flags
```

```
  $000b   11  rt_Version
  $000c   12  rt_Type
  $000d   13  rt_Pri
! $000e   14  rt_Name
! $0012   18  rt_IdString
! $0016   22  rt_Init
RexxArg:
  $0010   16  sizeof(RexxArg)
  $0000    0  ra_Size
  $0004    4  ra_Length
  $0006    6  ra_Flags
  $0007    7  ra_Hash
  $0008    8  ra_Buff[0]
RexxMsg:
  $0080  128  sizeof(RexxMsg)
  $0000    0  rm_Node
  $0014   20  rm_TaskBlock
  $0018   24  rm_LibBase
  $001c   28  rm_Action
  $0020   32  rm_Result1
  $0024   36  rm_Result2
  $0028   40  rm_Args[0]
  $0068  104  rm_PassPort
  $006c  108  rm_CommAddr
  $0070  112  rm_FileExt
  $0074  116  rm_Stdin
  $0078  120  rm_Stdout
  $007c  124  rm_avail
RexxMsgPort:
  $0050   80  sizeof(RexxMsgPort)
  $0000    0  rmp_Node
  $0020   32  rmp_Port
! $0042   66  rmp_ReplyList
RexxRsrc:
  $0020   32  sizeof(RexxRsrc)
  $0000    0  rr_Node
  $000e   14  rr_Func
  $0010   16  rr_Base
  $0014   20  rr_Size
  $0018   24  rr_Arg1
  $001c   28  rr_Arg2
RexxTask:
  $014a  330  sizeof(RexxTask)
  $0000    0  rt_Global[0]
  $00c8  200  rt_MsgPort
  $00ea  234  rt_Flags
  $00eb  235  rt_SigBit
  $00ec  236  rt_ClientID
  $00f0  240  rt_MsgPkt
  $00f4  244  rt_TaskID
  $00f8  248  rt_RexxPort
  $00fc  252  rt_ErrTrap
  $0100  256  rt_StackPtr
  $0104  260  rt_Header1
! $0112  274  rt_Header2
  $0120  288  rt_Header3
! $012e  302  rt_Header4
  $013c  316  rt_Header5
RigidDiskBlock:
  $0100  256  sizeof(RigidDiskBlock)
  $0000    0  rdb_ID
  $0004    4  rdb_SummedLongs
  $0008    8  rdb_ChkSum
  $000c   12  rdb_HostID
  $0010   16  rdb_BlockBytes
  $0014   20  rdb_Flags
```

Structure Reference

```
  $0018   24  rdb_BadBlockList
  $001c   28  rdb_PartitionList
  $0020   32  rdb_FileSysHeaderList
  $0024   36  rdb_DriveInit
  $0028   40  rdb_Reserved1[0]
  $0040   64  rdb_Cylinders
  $0044   68  rdb_Sectors
  $0048   72  rdb_Heads
  $004c   76  rdb_Interleave
  $0050   80  rdb_Park
  $0054   84  rdb_Reserved2[0]
  $0060   96  rdb_WritePreComp
  $0064  100  rdb_ReducedWrite
  $0068  104  rdb_StepRate
  $006c  108  rdb_Reserved3[0]
  $0080  128  rdb_RDBBlocksLo
  $0084  132  rdb_RDBBlocksHi
  $0088  136  rdb_LoCylinder
  $008c  140  rdb_HiCylinder
  $0090  144  rdb_CylBlocks
  $0094  148  rdb_AutoParkSeconds
  $0098  152  rdb_Reserved4[0]
  $00a0  160  rdb_DiskVendor[0]
  $00a8  168  rdb_DiskProduct[0]
  $00b8  184  rdb_DiskRevision[0]
  $00bc  188  rdb_ControllerVendor[0]
  $00c4  196  rdb_ControllerProduct[0]
  $00d4  212  rdb_ControllerRevision[0]
  $00d8  216  rdb_Reserved5[0]
RootNode:
  $0038   56  sizeof(RootNode)
  $0000    0  rn_TaskArray
  $0004    4  rn_ConsoleSegment
  $0008    8  rn_Time
  $0014   20  rn_RestartSeg
  $0018   24  rn_Info
  $001c   28  rn_FileHandlerSegment
  $0020   32  rn_CliList
  $002c   44  rn_BootProc
  $0030   48  rn_ShellSegment
  $0034   52  rn_Flags
RxsLib:
  $00fc  252  sizeof(RxsLib)
  $0000    0  rl_Node
  $0022   34  rl_Flags
  $0023   35  rl_Shadow
  $0024   36  rl_SysBase
  $0028   40  rl_DOSBase
  $002c   44  rl_IeeeDPBase
  $0030   48  rl_SegList
  $0034   52  rl_NIL
  $0038   56  rl_Chunk
  $003c   60  rl_MaxNest
  $0040   64  rl_NULL
  $0044   68  rl_FALSE
  $0048   72  rl_TRUE
  $004c   76  rl_REXX
  $0050   80  rl_COMMAND
  $0054   84  rl_STDIN
  $0058   88  rl_STDOUT
  $005c   92  rl_STDERR
  $0060   96  rl_Version
  $0064  100  rl_TaskName
  $0068  104  rl_TaskPri
  $006c  108  rl_TaskSeg
  $0070  112  rl_StackSize
```

```
  $0074  116  rl_RexxDir
  $0078  120  rl_CTABLE
  $007c  124  rl_Notice
  $0080  128  rl_RexxPort
  $00a2  162  rl_ReadLock
  $00a4  164  rl_TraceFH
  $00a8  168  rl_TaskList
  $00b6  182  rl_NumTask
  $00b8  184  rl_LibList
  $00c6  198  rl_NumLib
  $00c8  200  rl_ClipList
  $00d6  214  rl_NumClip
  $00d8  216  rl_MsgList
  $00e6  230  rl_NumMsg
  $00e8  232  rl_PgmList
  $00f6  246  rl_NumPgm
  $00f8  248  rl_TraceCnt
  $00fa  250  rl_avail
SCSICmd:
  $001e   30  sizeof(SCSICmd)
  $0000    0  scsi_Data
  $0004    4  scsi_Length
  $0008    8  scsi_Actual
  $000c   12  scsi_Command
  $0010   16  scsi_CmdLength
  $0012   18  scsi_CmdActual
  $0014   20  scsi_Flags
  $0015   21  scsi_Status
! $0016   22  scsi_SenseData
  $001a   26  scsi_SenseLength
  $001c   28  scsi_SenseActual
SGWork:
  $002c   44  sizeof(SGWork)
  $0000    0  Gadget
  $0004    4  StringInfo
  $0008    8  WorkBuffer
  $000c   12  PrevBuffer
  $0010   16  Modes
  $0014   20  IEvent
  $0018   24  Code
  $001a   26  BufferPos
  $001c   28  NumChars
! $001e   30  Actions
! $0022   34  LongInt
! $0026   38  GadgetInfo
  $002a   42  EditOp
SatisfyMsg:
  $001a   26  sizeof(SatisfyMsg)
  $0000    0  sm_Msg
  $0014   20  sm_Unit
! $0016   22  sm_ClipID
Screen:
  $015a  346  sizeof(Screen)
  $0000    0  NextScreen
  $0004    4  FirstWindow
  $0008    8  LeftEdge
  $000a   10  TopEdge
  $000c   12  Width
  $000e   14  Height
  $0010   16  MouseY
  $0012   18  MouseX
  $0014   20  Flags
! $0016   22  Title
! $001a   26  DefaultTitle
  $001e   30  BarHeight
  $001f   31  BarVBorder
```

Structure Reference

```
  $0020    32  BarHBorder
  $0021    33  MenuVBorder
  $0022    34  MenuHBorder
  $0023    35  WBorTop
  $0024    36  WBorLeft
  $0025    37  WBorRight
  $0026    38  WBorBottom
  $0028    40  Font
  $002c    44  ViewPort
  $0054    84  RastPort
  $00b8   184  BitMap
  $00e0   224  LayerInfo
! $0146   326  FirstGadget
  $014a   330  DetailPen
  $014b   331  BlockPen
  $014c   332  SaveColor0
! $014e   334  BarLayer
! $0152   338  ExtData
! $0156   342  UserData
Semaphore:
  $0024    36  sizeof(Semaphore)
  $0000     0  sm_MsgPort
  $0022    34  sm_Bids
SemaphoreRequest:
  $000c    12  sizeof(SemaphoreRequest)
  $0000     0  sr_Link
  $0008     8  sr_Waiter
SignalSemaphore:
  $002e    46  sizeof(SignalSemaphore)
  $0000     0  ss_Link
  $000e    14  ss_NestCount
  $0010    16  ss_WaitQueue
  $001c    28  ss_MultipleLink
  $0028    40  ss_Owner
  $002c    44  ss_QueueCount
SimpleSprite:
  $000c    12  sizeof(SimpleSprite)
  $0000     0  posctldata
  $0004     4  height
  $0006     6  x
  $0008     8  y
  $000a    10  num
SoftIntList:
  $0010    16  sizeof(SoftIntList)
  $0000     0  sh_List
  $000e    14  sh_Pad
SpecialMonitor:
  $003a    58  sizeof(SpecialMonitor)
  $0000     0  spm_Node
  $0018    24  spm_Flags
! $001a    26  do_monitor
! $001e    30  reserved1
! $0022    34  reserved2
! $0026    38  reserved3
! $002a    42  hblank
! $002e    46  vblank
! $0032    50  hsync
! $0036    54  vsync
SpriteDef:
  $0008     8  sizeof(SpriteDef)
  $0000     0  pos
  $0002     2  ctl
  $0004     4  dataa
  $0006     6  datab
SrcNode:
  $0010    16  sizeof(SrcNode)
```

```
  $0000     0  sn_Succ
  $0004     4  sn_Pred
  $0008     8  sn_Ptr
  $000c    12  sn_Size
StandardPacket:
  $0044    68  sizeof(StandardPacket)
  $0000     0  sp_Msg
  $0014    20  sp_Pkt
StoredProperty:
  $0008     8  sizeof(StoredProperty)
  $0000     0  sp_Size
  $0004     4  sp_Data
StringExtend:
  $0024    36  sizeof(StringExtend)
  $0000     0  Font
  $0004     4  Pens[0]
  $0006     6  ActivePens[0]
  $0008     8  InitialModes
  $000c    12  EditHook
  $0010    16  WorkBuffer
  $0014    20  Reserved[0]
StringInfo:
  $0024    36  sizeof(StringInfo)
  $0000     0  Buffer
  $0004     4  UndoBuffer
  $0008     8  BufferPos
  $000a    10  MaxChars
  $000c    12  DispPos
  $000e    14  UndoPos
  $0010    16  NumChars
  $0012    18  DispCount
  $0014    20  CLeft
  $0016    22  CTop
  $0018    24  Extension
  $001c    28  LongInt
  $0020    32  AltKeyMap
TAvailFonts:
  $000e    14  sizeof(TAvailFonts)
  $0000     0  taf_Type
! $0002     2  taf_Attr
TDU_PublicUnit:
  $0040    64  sizeof(TDU_PublicUnit)
  $0000     0  tdu_Unit
  $0026    38  tdu_Comp01Track
  $0028    40  tdu_Comp10Track
  $002a    42  tdu_Comp11Track
  $002c    44  tdu_StepDelay
  $0030    48  tdu_SettleDelay
  $0034    52  tdu_RetryCnt
  $0035    53  tdu_PubFlags
  $0036    54  tdu_CurrTrk
  $0038    56  tdu_CalibrateDelay
  $003c    60  tdu_Counter
TFontContents:
  $0104   260  sizeof(TFontContents)
  $0000     0  tfc_FileName[0]
  $00fe   254  tfc_TagCount
  $0100   256  tfc_YSize
  $0102   258  tfc_Style
  $0103   259  tfc_Flags
TTextAttr:
  $000c    12  sizeof(TTextAttr)
  $0000     0  tta_Name
  $0004     4  tta_YSize
  $0006     6  tta_Style
  $0007     7  tta_Flags
```

Structure Reference

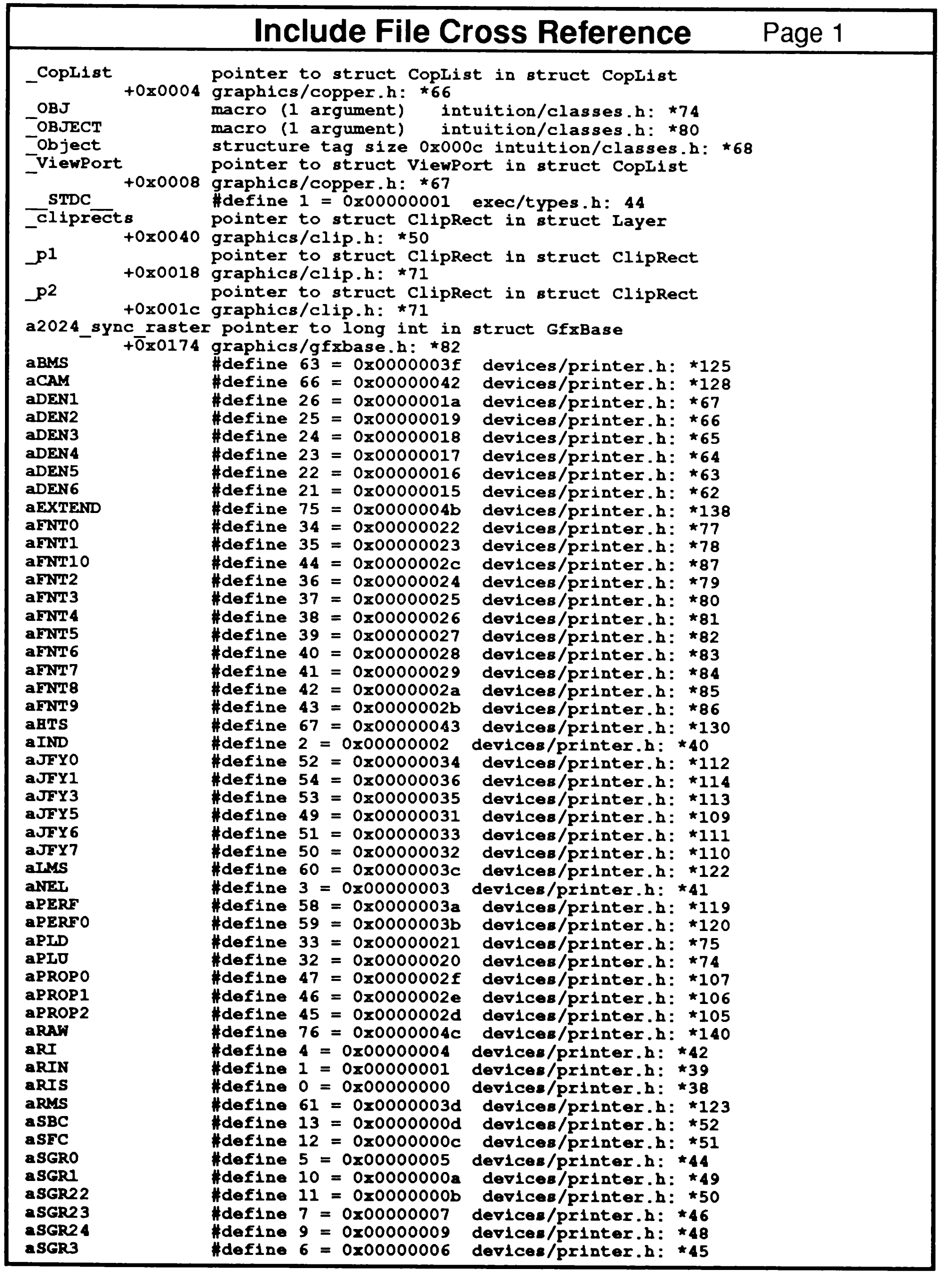
```
  $0008     8  tta_Tags
TagItem:
  $0008     8  sizeof(TagItem)
  $0000     0  ti_Tag
  $0004     4  ti_Data
Task:
  $005c    92  sizeof(Task)
  $0000     0  tc_Node
  $000e    14  tc_Flags
  $000f    15  tc_State
  $0010    16  tc_IDNestCnt
  $0011    17  tc_TDNestCnt
! $0012    18  tc_SigAlloc
! $0016    22  tc_SigWait
! $001a    26  tc_SigRecvd
! $001e    30  tc_SigExcept
  $0022    34  tc_TrapAlloc
  $0024    36  tc_TrapAble
! $0026    38  tc_ExceptData
! $002a    42  tc_ExceptCode
! $002e    46  tc_TrapData
! $0032    50  tc_TrapCode
! $0036    54  tc_SPReg
! $003a    58  tc_SPLower
! $003e    62  tc_SPUpper
! $0042    66  tc_Switch
! $0046    70  tc_Launch
! $004a    74  tc_MemEntry
  $0058    88  tc_UserData
TextAttr:
  $0008     8  sizeof(TextAttr)
  $0000     0  ta_Name
  $0004     4  ta_YSize
  $0006     6  ta_Style
  $0007     7  ta_Flags
TextExtent:
  $000c    12  sizeof(TextExtent)
  $0000     0  te_Width
  $0002     2  te_Height
  $0004     4  te_Extent
TextFont:
  $0034    52  sizeof(TextFont)
  $0000     0  tf_Message
  $0014    20  tf_YSize
  $0016    22  tf_Style
  $0017    23  tf_Flags
  $0018    24  tf_XSize
  $001a    26  tf_Baseline
  $001c    28  tf_BoldSmear
  $001e    30  tf_Accessors
  $0020    32  tf_LoChar
  $0021    33  tf_HiChar
! $0022    34  tf_CharData
  $0026    38  tf_Modulo
  $0028    40  tf_CharLoc
  $002c    44  tf_CharSpace
  $0030    48  tf_CharKern
TextFontExtension:
  $0018    24  sizeof(TextFontExtension)
  $0000     0  tfe_MatchWord
  $0002     2  tfe_Flags0
  $0003     3  tfe_Flags1
  $0004     4  tfe_BackPtr
  $0008     8  tfe_OrigReplyPort
  $000c    12  tfe_Tags
  $0010    16  tfe_OFontPatchS
```

```
  $0014    20  tfe_OFontPatchK
TmpRas:
  $0008     8  sizeof(TmpRas)
  $0000     0  RasPtr
  $0004     4  Size
UCopList:
  $000c    12  sizeof(UCopList)
  $0000     0  Next
  $0004     4  FirstCopList
  $0008     8  CopList
Unit:
  $0026    38  sizeof(Unit)
  $0000     0  unit_MsgPort
  $0022    34  unit_flags
  $0023    35  unit_pad
  $0024    36  unit_OpenCnt
VSprite:
  $003c    60  sizeof(VSprite)
  $0000     0  NextVSprite
  $0004     4  PrevVSprite
  $0008     8  DrawPath
  $000c    12  ClearPath
  $0010    16  OldY
  $0012    18  OldX
  $0014    20  Flags
  $0016    22  Y
  $0018    24  X
  $001a    26  Height
  $001c    28  Width
  $001e    30  Depth
  $0020    32  MeMask
  $0022    34  HitMask
  $0024    36  ImageData
  $0028    40  BorderLine
  $002c    44  CollMask
  $0030    48  SprColors
  $0034    52  VSBob
  $0038    56  PlanePick
  $0039    57  PlaneOnOff
  $003a    58  VUserExt
View:
  $0012    18  sizeof(View)
  $0000     0  ViewPort
  $0004     4  LOFCprList
  $0008     8  SHFCprList
  $000c    12  DyOffset
  $000e    14  DxOffset
  $0010    16  Modes
ViewExtra:
  $0020    32  sizeof(ViewExtra)
  $0000     0  n
  $0018    24  View
  $001c    28  Monitor
ViewPort:
  $0028    40  sizeof(ViewPort)
  $0000     0  Next
  $0004     4  ColorMap
  $0008     8  DspIns
  $000c    12  SprIns
  $0010    16  ClrIns
  $0014    20  UCopIns
  $0018    24  DWidth
  $001a    26  DHeight
  $001c    28  DxOffset
  $001e    30  DyOffset
  $0020    32  Modes
  $0022   34  SpritePriorities
  $0023   35  ExtendedModes
  $0024   36  RasInfo
ViewPortExtra:
  $0024   36  sizeof(ViewPortExtra)
  $0000    0  n
  $0018   24  ViewPort
  $001c   28  DisplayClip
WBArg:
  $0008    8  sizeof(WBArg)
  $0000    0  wa_Lock
  $0004    4  wa_Name
WBStartup:
  $0028   40  sizeof(WBStartup)
  $0000    0  sm_Message
  $0014   20  sm_Process
  $0018   24  sm_Segment
  $001c   28  sm_NumArgs
  $0020   32  sm_ToolWindow
  $0024   36  sm_ArgList
Window:
  $0088  136  sizeof(Window)
  $0000    0  NextWindow
  $0004    4  LeftEdge
  $0006    6  TopEdge
  $0008    8  Width
  $000a   10  Height
  $000c   12  MouseY
  $000e   14  MouseX
  $0010   16  MinWidth
  $0012   18  MinHeight
  $0014   20  MaxWidth
  $0016   22  MaxHeight
  $0018   24  Flags
  $001c   28  MenuStrip
  $0020   32  Title
  $0024   36  FirstRequest
  $0028   40  DMRequest
  $002c   44  ReqCount
! $002e   46  WScreen
! $0032   50  RPort
  $0036   54  BorderLeft
  $0037   55  BorderTop
  $0038   56  BorderRight
  $0039   57  BorderBottom
! $003a   58  BorderRPort
! $003e   62  FirstGadget
! $0042   66  Parent
! $0046   70  Descendant
! $004a   74  Pointer
  $004e   78  PtrHeight
  $004f   79  PtrWidth
  $0050   80  XOffset
  $0051   81  YOffset
! $0052   82  IDCMPFlags
! $0056   86  UserPort
! $005a   90  WindowPort
! $005e   94  MessageKey
  $0062   98  DetailPen
  $0063   99  BlockPen
  $0064  100  CheckMark
  $0068  104  ScreenTitle
  $006c  108  GZZMouseX
  $006e  110  GZZMouseY
  $0070  112  GZZWidth
  $0072  114  GZZHeight
```

```
  $0074  116  ExtData
  $0078  120  UserData
  $007c  124  WLayer
  $0080  128  IFont
  $0084  132  MoreFlags
_Object:
  $000c   12  sizeof(_Object)
  $0000    0  o_Node
  $0008    8  o_Class
bltnode:
  $0012   18  sizeof(bltnode)
  $0000    0  n
  $0004    4  function
  $0008    8  stat
  $000a   10  blitsize
  $000c   12  beamsync
! $000e   14  cleanup
collTable:
  $0040   64  sizeof(collTable)
  $0000    0  collPtrs[0]
copinit:
  $0078  120  sizeof(copinit)
  $0000    0  vsync_hblank[0]
  $0004    4  diwstart[0]
  $000c   12  diagstrt[0]
  $0014   20  sprstrtup[0]
  $0054   84  wait14[0]
  $0058   88  norm_hblank[0]
  $005c   92  genloc[0]
  $0064  100  jump[0]
  $006c  108  wait_forever[0]
  $0070  112  sprstop[0]
cprlist:
  $000a   10  sizeof(cprlist)
  $0000    0  Next
  $0004    4  start
  $0008    8  MaxCount
gpGoInactive:
  $0008    8  sizeof(gpGoInactive)
  $0000    0  MethodID
  $0004    4  gpgi_GInfo
gpHitTest:
  $000c   12  sizeof(gpHitTest)
  $0000    0  MethodID
  $0004    4  gpht_GInfo
  $0008    8  gpht_Mouse
  $0008    8  gpht_Mouse.X
  $000a   10  gpht_Mouse.Y
gpInput:
  $0014   20  sizeof(gpInput)
  $0000    0  MethodID
  $0004    4  gpi_GInfo
  $0008    8  gpi_IEvent
  $000c   12  gpi_Termination
  $0010   16  gpi_Mouse
  $0010   16  gpi_Mouse.X
  $0012   18  gpi_Mouse.Y
gpRender:
  $0010   16  sizeof(gpRender)
  $0000    0  MethodID
  $0004    4  gpr_GInfo
  $0008    8  gpr_RPort
  $000c   12  gpr_Redraw
impDraw:
  $0018   24  sizeof(impDraw)
  $0000    0  MethodID
```

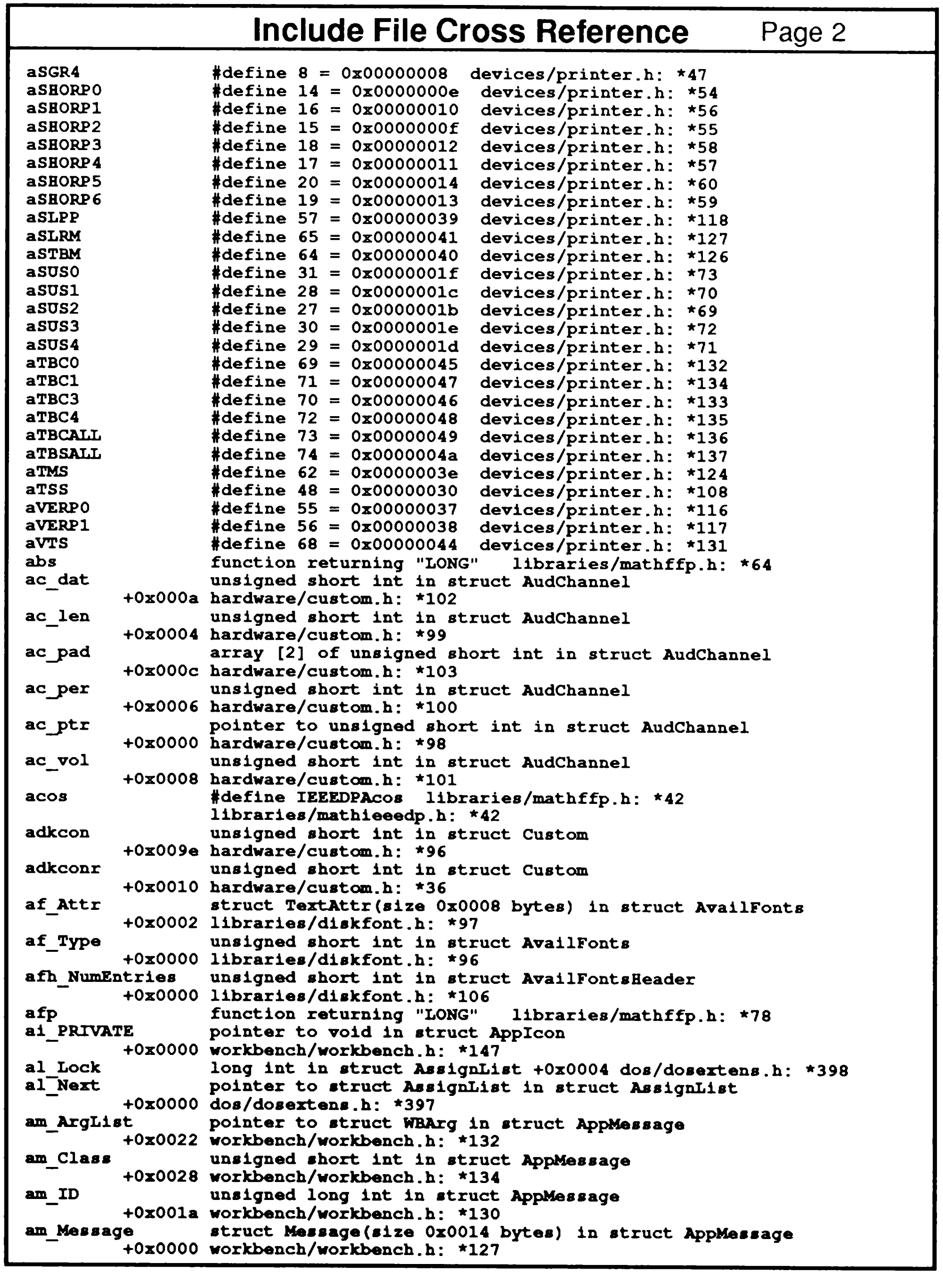
```
  $0004    4  imp_RPort
  $0008    8  imp_Offset
  $0008    8  imp_Offset.X
  $000a   10  imp_Offset.Y
  $000c   12  imp_State
  $0010   16  imp_DrInfo
  $0014   20  imp_Dimensions
  $0014   20  imp_Dimensions.Width
  $0016   22  imp_Dimensions.Height
impErase:
  $0010   16  sizeof(impErase)
  $0000    0  MethodID
  $0004    4  imp_RPort
  $0008    8  imp_Offset
  $0008    8  imp_Offset.X
  $000a   10  imp_Offset.Y
  $000c   12  imp_Dimensions
  $000c   12  imp_Dimensions.Width
  $000e   14  imp_Dimensions.Height
impFrameBox:
  $0014   20  sizeof(impFrameBox)
  $0000    0  MethodID
  $0004    4  imp_ContentsBox
  $0008    8  imp_FrameBox
  $000c   12  imp_DrInfo
  $0010   16  imp_FrameFlags
impHitTest:
  $000c   12  sizeof(impHitTest)
  $0000    0  MethodID
  $0004    4  imp_Point
  $0004    4  imp_Point.X
  $0006    6  imp_Point.Y
  $0008    8  imp_Dimensions
  $0008    8  imp_Dimensions.Width
  $000a   10  imp_Dimensions.Height
mouth_rb:
  $005c   92  sizeof(mouth_rb)
  $0000    0  voice
  $0058   88  width
  $0059   89  height
  $005a   90  shape
  $005b   91  sync
narrator_rb:
  $0058   88  sizeof(narrator_rb)
  $0000    0  message
  $0030   48  rate
  $0032   50  pitch
  $0034   52  mode
  $0036   54  sex
  $0038   56  ch_masks
  $003c   60  nm_masks
  $003e   62  volume
  $0040   64  sampfreq
  $0042   66  mouths
  $0043   67  chanmask
```

```
  $0044   68  numchan
  $0045   69  flags
  $0046   70  F0enthusiasm
  $0047   71  F0perturb
  $0048   72  F1adj
  $0049   73  F2adj
  $004a   74  F3adj
  $004b   75  A1adj
  $004c   76  A2adj
  $004d   77  A3adj
  $004e   78  articulate
  $004f   79  centralize
  $0050   80  centphon
  $0054   84  AVbias
  $0055   85  AFbias
  $0056   86  priority
  $0057   87  pad1
opAddTail:
  $0008    8  sizeof(opAddTail)
  $0000    0  MethodID
  $0004    4  opat_List
opGet:
  $000c   12  sizeof(opGet)
  $0000    0  MethodID
  $0004    4  opg_AttrID
  $0008    8  opg_Storage
opMember:
  $0008    8  sizeof(opMember)
  $0000    0  MethodID
  $0004    4  opam_Object
opSet:
  $000c   12  sizeof(opSet)
  $0000    0  MethodID
  $0004    4  ops_AttrList
  $0008    8  ops_GInfo
opUpdate:
  $0010   16  sizeof(opUpdate)
  $0000    0  MethodID
  $0004    4  opu_AttrList
  $0008    8  opu_GInfo
  $000c   12  opu_Flags
timerequest:
  $0028   40  sizeof(timerequest)
  $0000    0  tr_node
  $0020   32  tr_time
timeval:
  $0008    8  sizeof(timeval)
  $0000    0  tv_secs
  $0004    4  tv_micro
colorEntry:
  $0004    4  sizeof(colorEntry)
  $0000    0  colorLong
  $0000    0  colorByte[0]
  $0000    0  colorSByte[0]
```



Include File Cross Reference Page 1

```
_CopList          pointer to struct CopList in struct CopList
          +0x0004 graphics/copper.h: *66
_OBJ              macro (1 argument)   intuition/classes.h: *74
_OBJECT           macro (1 argument)   intuition/classes.h: *80
_Object           structure tag size 0x000c intuition/classes.h: *68
_ViewPort         pointer to struct ViewPort in struct CopList
          +0x0008 graphics/copper.h: *67
__STDC__          #define 1 = 0x00000001  exec/types.h: 44
_cliprects        pointer to struct ClipRect in struct Layer
          +0x0040 graphics/clip.h: *50
_p1               pointer to struct ClipRect in struct ClipRect
          +0x0018 graphics/clip.h: *71
_p2               pointer to struct ClipRect in struct ClipRect
          +0x001c graphics/clip.h: *71
a2024_sync_raster pointer to long int in struct GfxBase
          +0x0174 graphics/gfxbase.h: *82
aBMS              #define 63 = 0x0000003f  devices/printer.h: *125
aCAM              #define 66 = 0x00000042  devices/printer.h: *128
aDEN1             #define 26 = 0x0000001a  devices/printer.h: *67
aDEN2             #define 25 = 0x00000019  devices/printer.h: *66
aDEN3             #define 24 = 0x00000018  devices/printer.h: *65
aDEN4             #define 23 = 0x00000017  devices/printer.h: *64
aDEN5             #define 22 = 0x00000016  devices/printer.h: *63
aDEN6             #define 21 = 0x00000015  devices/printer.h: *62
aEXTEND           #define 75 = 0x0000004b  devices/printer.h: *138
aFNT0             #define 34 = 0x00000022  devices/printer.h: *77
aFNT1             #define 35 = 0x00000023  devices/printer.h: *78
aFNT10            #define 44 = 0x0000002c  devices/printer.h: *87
aFNT2             #define 36 = 0x00000024  devices/printer.h: *79
aFNT3             #define 37 = 0x00000025  devices/printer.h: *80
aFNT4             #define 38 = 0x00000026  devices/printer.h: *81
aFNT5             #define 39 = 0x00000027  devices/printer.h: *82
aFNT6             #define 40 = 0x00000028  devices/printer.h: *83
aFNT7             #define 41 = 0x00000029  devices/printer.h: *84
aFNT8             #define 42 = 0x0000002a  devices/printer.h: *85
aFNT9             #define 43 = 0x0000002b  devices/printer.h: *86
aHTS              #define 67 = 0x00000043  devices/printer.h: *130
aIND              #define 2 = 0x00000002  devices/printer.h: *40
aJFY0             #define 52 = 0x00000034  devices/printer.h: *112
aJFY1             #define 54 = 0x00000036  devices/printer.h: *114
aJFY3             #define 53 = 0x00000035  devices/printer.h: *113
aJFY5             #define 49 = 0x00000031  devices/printer.h: *109
aJFY6             #define 51 = 0x00000033  devices/printer.h: *111
aJFY7             #define 50 = 0x00000032  devices/printer.h: *110
aLMS              #define 60 = 0x0000003c  devices/printer.h: *122
aNEL              #define 3 = 0x00000003  devices/printer.h: *41
aPERF             #define 58 = 0x0000003a  devices/printer.h: *119
aPERF0            #define 59 = 0x0000003b  devices/printer.h: *120
aPLD              #define 33 = 0x00000021  devices/printer.h: *75
aPLU              #define 32 = 0x00000020  devices/printer.h: *74
aPROP0            #define 47 = 0x0000002f  devices/printer.h: *107
aPROP1            #define 46 = 0x0000002e  devices/printer.h: *106
aPROP2            #define 45 = 0x0000002d  devices/printer.h: *105
aRAW              #define 76 = 0x0000004c  devices/printer.h: *140
aRI               #define 4 = 0x00000004  devices/printer.h: *42
aRIN              #define 1 = 0x00000001  devices/printer.h: *39
aRIS              #define 0 = 0x00000000  devices/printer.h: *38
aRMS              #define 61 = 0x0000003d  devices/printer.h: *123
aSBC              #define 13 = 0x0000000d  devices/printer.h: *52
aSFC              #define 12 = 0x0000000c  devices/printer.h: *51
aSGR0             #define 5 = 0x00000005  devices/printer.h: *44
aSGR1             #define 10 = 0x0000000a  devices/printer.h: *49
aSGR22            #define 11 = 0x0000000b  devices/printer.h: *50
aSGR23            #define 7 = 0x00000007  devices/printer.h: *46
aSGR24            #define 9 = 0x00000009  devices/printer.h: *48
aSGR3             #define 6 = 0x00000006  devices/printer.h: *45
```



Include File Cross Reference Page 2

```
aSGR4             #define 8 = 0x00000008  devices/printer.h: *47
aSHORP0           #define 14 = 0x0000000e  devices/printer.h: *54
aSHORP1           #define 16 = 0x00000010  devices/printer.h: *56
aSHORP2           #define 15 = 0x0000000f  devices/printer.h: *55
aSHORP3           #define 18 = 0x00000012  devices/printer.h: *58
aSHORP4           #define 17 = 0x00000011  devices/printer.h: *57
aSHORP5           #define 20 = 0x00000014  devices/printer.h: *60
aSHORP6           #define 19 = 0x00000013  devices/printer.h: *59
aSLPP             #define 57 = 0x00000039  devices/printer.h: *118
aSLRM             #define 65 = 0x00000041  devices/printer.h: *127
aSTBM             #define 64 = 0x00000040  devices/printer.h: *126
aSUS0             #define 31 = 0x0000001f  devices/printer.h: *73
aSUS1             #define 28 = 0x0000001c  devices/printer.h: *70
aSUS2             #define 27 = 0x0000001b  devices/printer.h: *69
aSUS3             #define 30 = 0x0000001e  devices/printer.h: *72
aSUS4             #define 29 = 0x0000001d  devices/printer.h: *71
aTBC0             #define 69 = 0x00000045  devices/printer.h: *132
aTBC1             #define 71 = 0x00000047  devices/printer.h: *134
aTBC3             #define 70 = 0x00000046  devices/printer.h: *133
aTBC4             #define 72 = 0x00000048  devices/printer.h: *135
aTBCALL           #define 73 = 0x00000049  devices/printer.h: *136
aTBSALL           #define 74 = 0x0000004a  devices/printer.h: *137
aTMS              #define 62 = 0x0000003e  devices/printer.h: *124
aTSS              #define 48 = 0x00000030  devices/printer.h: *108
aVERP0            #define 55 = 0x00000037  devices/printer.h: *116
aVERP1            #define 56 = 0x00000038  devices/printer.h: *117
aVTS              #define 68 = 0x00000044  devices/printer.h: *131
abs               function returning "LONG"   libraries/mathffp.h: *64
ac_dat            unsigned short int in struct AudChannel
          +0x000a hardware/custom.h: *102
ac_len            unsigned short int in struct AudChannel
          +0x0004 hardware/custom.h: *99
ac_pad            array [2] of unsigned short int in struct AudChannel
          +0x000c hardware/custom.h: *103
ac_per            unsigned short int in struct AudChannel
          +0x0006 hardware/custom.h: *100
ac_ptr            pointer to unsigned short int in struct AudChannel
          +0x0000 hardware/custom.h: *98
ac_vol            unsigned short int in struct AudChannel
          +0x0008 hardware/custom.h: *101
acos              #define IEEEDPAcos  libraries/mathffp.h: *42
                  libraries/mathieeedp.h: *42
adkcon            unsigned short int in struct Custom
          +0x009e hardware/custom.h: *96
adkconr           unsigned short int in struct Custom
          +0x0010 hardware/custom.h: *36
af_Attr           struct TextAttr(size 0x0008 bytes) in struct AvailFonts
          +0x0002 libraries/diskfont.h: *97
af_Type           unsigned short int in struct AvailFonts
          +0x0000 libraries/diskfont.h: *96
afh_NumEntries    unsigned short int in struct AvailFontsHeader
          +0x0000 libraries/diskfont.h: *106
afp               function returning "LONG"   libraries/mathffp.h: *78
ai_PRIVATE        pointer to void in struct AppIcon
          +0x0000 workbench/workbench.h: *147
al_Lock           long int in struct AssignList +0x0004 dos/dosextens.h: *398
al_Next           pointer to struct AssignList in struct AssignList
          +0x0000 dos/dosextens.h: *397
am_ArgList        pointer to struct WBArg in struct AppMessage
          +0x0022 workbench/workbench.h: *132
am_Class          unsigned short int in struct AppMessage
          +0x0028 workbench/workbench.h: *134
am_ID             unsigned long int in struct AppMessage
          +0x001a workbench/workbench.h: *130
am_Message        struct Message(size 0x0014 bytes) in struct AppMessage
          +0x0000 workbench/workbench.h: *127
```

Include File Cross Reference Page 3

```
am_Micros        unsigned long int in struct AppMessage
         +0x0032 workbench/workbench.h: *138
am_MouseX        short int in struct AppMessage
         +0x002a workbench/workbench.h: *135
am_MouseY        short int in struct AppMessage
         +0x002c workbench/workbench.h: *136
am_NumArgs       long int in struct AppMessage
         +0x001e workbench/workbench.h: *131
am_Reserved      array [8] of unsigned long int in struct AppMessage
         +0x0036 workbench/workbench.h: *139
am_Seconds       unsigned long int in struct AppMessage
         +0x002e workbench/workbench.h: *137
am_Type          unsigned short int in struct AppMessage
         +0x0014 workbench/workbench.h: *128
am_UserData      unsigned long int in struct AppMessage
         +0x0016 workbench/workbench.h: *129
am_Version       unsigned short int in struct AppMessage
         +0x0026 workbench/workbench.h: *133
ami_PRIVATE      pointer to void in struct AppMenuItem
         +0x0000 workbench/workbench.h: *148
an_Child         pointer to struct AChain in struct AChain
         +0x0000 dos/dosasl.h: *99
an_Flags         char in struct AChain +0x0110 dos/dosasl.h: *103
an_Info          struct FileInfoBlock(size 0x0104 bytes) in struct AChain
         +0x000c dos/dosasl.h: *102
an_Lock          long int in struct AChain +0x0008 dos/dosasl.h: *101
an_Parent        pointer to struct AChain in struct AChain
         +0x0004 dos/dosasl.h: *100
an_String        array [1] of unsigned char in struct AChain
         +0x0111 dos/dosasl.h: *104
ap_Base          pointer to struct AChain in struct AnchorPath
         +0x0000 dos/dosasl.h: *55
ap_BreakBits     long int in struct AnchorPath +0x0008 dos/dosasl.h: *59
ap_Buf           array [1] of unsigned char in struct AnchorPath
         +0x0118 dos/dosasl.h: *66
ap_Current       #define ap_Last  dos/dosasl.h: *58
ap_First         #define ap_Base  dos/dosasl.h: *56
ap_Flags         char in struct AnchorPath +0x0010 dos/dosasl.h: *61
ap_FoundBreak    long int in struct AnchorPath +0x000c dos/dosasl.h: *60
ap_Info          struct FileInfoBlock(size 0x0104 bytes) in struct AnchorPath
         +0x0014 dos/dosasl.h: *65
ap_Last          pointer to struct AChain in struct AnchorPath
         +0x0004 dos/dosasl.h: *57
ap_Length        #define ap_Flags  dos/dosasl.h: *64
ap_Reserved      char in struct AnchorPath +0x0011 dos/dosasl.h: *62
ap_Strlen        short int in struct AnchorPath +0x0012 dos/dosasl.h: *63
articulate       unsigned char in struct narrator_rb
         +0x004e devices/narrator.h: *115
asi_Start        unsigned short int in struct AnalogSignalInterval
         +0x0000 graphics/monitor.h: *138
asi_Stop         unsigned short int in struct AnalogSignalInterval
         +0x0002 graphics/monitor.h: *139
asin             #define IEEEDPAsin  libraries/mathffp.h: *44
                 libraries/mathieeedp.h: *44
atan             #define IEEEDPAtan  libraries/mathffp.h: *40
                 libraries/mathieeedp.h: *40
aud              array [4] of struct AudChannel(size 0x0010 bytes) in struct
                 Custom
         +0x00a0 hardware/custom.h: *104
aw_PRIVATE       pointer to void in struct AppWindow
         +0x0000 workbench/workbench.h: *146
Aladj            char in struct narrator_rb +0x004b devices/narrator.h: *112
A2024FIFTEENHERTZ_KEY #define 0x00049000 = 0x00049000
                 graphics/displayinfo.h: *210
A2024TENHERTZ_KEY #define 0x00041000 = 0x00041000
                 graphics/displayinfo.h: *209
```

Include File Cross Reference Page 4

```
A2024_MONITOR_ID #define 0x00041000 = 0x00041000  graphics/displayinfo.h: *207
A2adj            char in struct narrator_rb +0x004c devices/narrator.h: *113
A3adj            char in struct narrator_rb +0x004d devices/narrator.h: *114
ABC              #define 0x80 = 0x00000080  hardware/blit.h: *32
ABNC             #define 0x40 = 0x00000040  hardware/blit.h: *33
ABORT_BUSY       #define 288 = 0x00000120  dos/dosextens.h: *460
ABORT_DISK_ERROR #define 296 = 0x00000128  dos/dosextens.h: *459
ABS              macro (1 argument)   clib/macros.h: *17
ABSOLUTE_DIMENSIONS #define 0x0020 = 0x00000020  intuition/preferences.h: *249
ACCESS_READ      #define -2 = 0xfffffffe  dos/dos.h: *50
ACCESS_WRITE     #define -1 = 0xffffffff  dos/dos.h: *52
ACTION_ADD_NOTIFY #define 4097 = 0x00001001  dos/dosextens.h: *211
ACTION_CHANGE_MODE #define 1028 = 0x00000404  dos/dosextens.h: *201
ACTION_CHANGE_SIGNAL #define 995 = 0x000003e3  dos/dosextens.h: *193
ACTION_COPY_DIR  #define 19 = 0x00000013  dos/dosextens.h: *162
ACTION_COPY_DIR_FH #define 1030 = 0x00000406  dos/dosextens.h: *203
ACTION_CREATE_DIR #define 22 = 0x00000016  dos/dosextens.h: *165
ACTION_CURRENT_VOLUME #define 7 = 0x00000007  dos/dosextens.h: *153
ACTION_DELETE_OBJECT #define 16 = 0x00000010  dos/dosextens.h: *159
ACTION_DIE       #define 5 = 0x00000005  dos/dosextens.h: *151
ACTION_DISK_CHANGE #define 33 = 0x00000021  dos/dosextens.h: *176
ACTION_DISK_INFO #define 25 = 0x00000019  dos/dosextens.h: *168
ACTION_DISK_TYPE #define 32 = 0x00000020  dos/dosextens.h: *175
ACTION_END       #define 1007 = 0x000003ef  dos/dosextens.h: *187
ACTION_EVENT     #define 6 = 0x00000006  dos/dosextens.h: *152
ACTION_EXAMINE_ALL #define 1033 = 0x00000409  dos/dosextens.h: *205
ACTION_EXAMINE_FH #define 1034 = 0x0000040a  dos/dosextens.h: *206
ACTION_EXAMINE_NEXT #define 24 = 0x00000018  dos/dosextens.h: *167
ACTION_EXAMINE_OBJECT #define 23 = 0x00000017  dos/dosextens.h: *166
ACTION_FH_FROM_LOCK #define 1026 = 0x00000402  dos/dosextens.h: *199
ACTION_FINDINPUT #define 1005 = 0x000003ed  dos/dosextens.h: *185
ACTION_FINDOUTPUT #define 1006 = 0x000003ee  dos/dosextens.h: *186
ACTION_FINDUPDATE #define 1004 = 0x000003ec  dos/dosextens.h: *184
ACTION_FLUSH     #define 27 = 0x0000001b  dos/dosextens.h: *170
ACTION_FORMAT    #define 1020 = 0x000003fc  dos/dosextens.h: *194
ACTION_FREE_LOCK #define 15 = 0x0000000f  dos/dosextens.h: *158
ACTION_FREE_RECORD #define 2009 = 0x000007d9  dos/dosextens.h: *209
ACTION_GET_BLOCK #define 2 = 0x00000002  dos/dosextens.h: *149
ACTION_INFO      #define 26 = 0x0000001a  dos/dosextens.h: *169
ACTION_INHIBIT   #define 31 = 0x0000001f  dos/dosextens.h: *174
ACTION_IS_FILESYSTEM #define 1027 = 0x00000403  dos/dosextens.h: *200
ACTION_LOCATE_OBJECT #define 8 = 0x00000008  dos/dosextens.h: *154
ACTION_LOCK_RECORD #define 2008 = 0x000007d8  dos/dosextens.h: *208
ACTION_MAKE_LINK #define 1021 = 0x000003fd  dos/dosextens.h: *195
ACTION_MORE_CACHE #define 18 = 0x00000012  dos/dosextens.h: *161
ACTION_NIL       #define 0 = 0x00000000  dos/dosextens.h: *147
ACTION_PARENT    #define 29 = 0x0000001d  dos/dosextens.h: *172
ACTION_PARENT_FH #define 1031 = 0x00000407  dos/dosextens.h: *204
ACTION_QUEUE     #define 2003L = 0x000007d3  rexx/rexxio.h: *81
ACTION_READ      #define 'R' = 0x00000052  dos/dosextens.h: *157
ACTION_READ_LINK #define 1024 = 0x00000400  dos/dosextens.h: *198
ACTION_READ_RETURN #define 1001 = 0x000003e9  dos/dosextens.h: *181
ACTION_REMOVE_NOTIFY #define 4098 = 0x00001002  dos/dosextens.h: *212
ACTION_RENAME_DISK #define 9 = 0x00000009  dos/dosextens.h: *155
ACTION_RENAME_OBJECT #define 17 = 0x00000011  dos/dosextens.h: *160
ACTION_SAME_LOCK #define 40 = 0x00000028  dos/dosextens.h: *192
ACTION_SCREEN_MODE #define 994 = 0x000003e2  dos/dosextens.h: *179
ACTION_SEEK      #define 1008 = 0x000003f0  dos/dosextens.h: *183
ACTION_SET_COMMENT #define 28 = 0x0000001c  dos/dosextens.h: *171
ACTION_SET_DATE  #define 34 = 0x00000022  dos/dosextens.h: *177
ACTION_SET_FILE_SIZE #define 1022 = 0x000003fe  dos/dosextens.h: *188
ACTION_SET_MAP   #define 4 = 0x00000004  dos/dosextens.h: *150
ACTION_SET_PROTECT #define 21 = 0x00000015  dos/dosextens.h: *164
ACTION_STACK     #define 2002L = 0x000007d2  rexx/rexxio.h: *80
ACTION_STARTUP   #define 0 = 0x00000000  dos/dosextens.h: *148
ACTION_TIMER     #define 30 = 0x0000001e  dos/dosextens.h: *173
```

Include File Cross Reference Page 5

```
ACTION_WAIT_CHAR #define 20 = 0x00000014  dos/dosextens.h: *163
ACTION_WRITE     #define 'W' = 0x00000057  dos/dosextens.h: *156
ACTION_WRITE_PROTECT #define 1023 = 0x000003ff  dos/dosextens.h: *189
ACTION_WRITE_RETURN #define 1002 = 0x000003ea  dos/dosextens.h: *182
ACTIVATE         #define WFLG_ACTIVATE = 0x00001000
                 intuition/iobsolete.h: *160
ACTIVEGADGET     #define GACT_ACTIVEGADGET = 0x00004000
                 intuition/iobsolete.h: *86
ACTIVEWINDOW     #define IDCMP_ACTIVEWINDOW = 0x00040000
                 intuition/iobsolete.h: *132
AChain           structure tag size 0x0112 dos/dosasl.h: *55, 57, 98, 99, 100
ADALLOC_MAXPREC  #define 127 = 0x0000007f  devices/audio.h: *24
ADALLOC_MINPREC  #define -128 = 0xffffff80  devices/audio.h: *23
ADCMD_ALLOCATE   #define 32 = 0x00000020  devices/audio.h: *32
ADCMD_FINISH     #define (CMD_NONSTD+2) = 0x0000000b  devices/audio.h: *28
ADCMD_FREE       #define (CMD_NONSTD+0) = 0x00000009  devices/audio.h: *26
ADCMD_LOCK       #define (CMD_NONSTD+4) = 0x0000000d  devices/audio.h: *30
ADCMD_PERVOL     #define (CMD_NONSTD+3) = 0x0000000c  devices/audio.h: *29
ADCMD_SETPREC    #define (CMD_NONSTD+1) = 0x0000000a  devices/audio.h: *27
ADCMD_WAITCYCLE  #define (CMD_NONSTD+5) = 0x0000000e  devices/audio.h: *31
ADHARD_CHANNELS  #define 4 = 0x00000004  devices/audio.h: *21
ADIOB_NOWAIT     #define 6 = 0x00000006  devices/audio.h: *38
ADIOB_PERVOL     #define 4 = 0x00000004  devices/audio.h: *34
ADIOB_SYNCCYCLE  #define 5 = 0x00000005  devices/audio.h: *36
ADIOB_WRITEMESSAGE #define 7 = 0x00000007  devices/audio.h: *40
ADIOERR_ALLOCFAILED #define -11 = 0xfffffff5  devices/audio.h: *44
ADIOERR_CHANNELSTOLEN #define -12 = 0xfffffff4  devices/audio.h: *45
ADIOERR_NOALLOCATION #define -10 = 0xfffffff6  devices/audio.h: *43
ADIOF_NOWAIT     #define (1<<6) = 0x00000040  devices/audio.h: *39
ADIOF_PERVOL     #define (1<<4) = 0x00000010  devices/audio.h: *35
ADIOF_SYNCCYCLE  #define (1<<5) = 0x00000020  devices/audio.h: *37
ADIOF_WRITEMESSAGE #define (1<<7) = 0x00000080  devices/audio.h: *41
ADKB_FAST        #define 8 = 0x00000008  hardware/adkbits.h: *22
ADKB_MFMPREC     #define 12 = 0x0000000c  hardware/adkbits.h: *18
ADKB_MSBSYNC     #define 9 = 0x00000009  hardware/adkbits.h: *21
ADKB_PRECOMP0    #define 13 = 0x0000000d  hardware/adkbits.h: *17
ADKB_PRECOMP1    #define 14 = 0x0000000e  hardware/adkbits.h: *16
ADKB_SETCLR      #define 15 = 0x0000000f  hardware/adkbits.h: *15
ADKB_UARTBRK     #define 11 = 0x0000000b  hardware/adkbits.h: *19
ADKB_USE0P1      #define 4 = 0x00000004  hardware/adkbits.h: *26
ADKB_USE0V1      #define 0 = 0x00000000  hardware/adkbits.h: *30
ADKB_USE1P2      #define 5 = 0x00000005  hardware/adkbits.h: *25
ADKB_USE1V2      #define 1 = 0x00000001  hardware/adkbits.h: *29
ADKB_USE2P3      #define 6 = 0x00000006  hardware/adkbits.h: *24
ADKB_USE2V3      #define 2 = 0x00000002  hardware/adkbits.h: *28
ADKB_USE3PN      #define 7 = 0x00000007  hardware/adkbits.h: *23
ADKB_USE3VN      #define 3 = 0x00000003  hardware/adkbits.h: *27
ADKB_WORDSYNC    #define 10 = 0x0000000a  hardware/adkbits.h: *20
ADKF_FAST        #define (1<<8) = 0x00000100  hardware/adkbits.h: *39
ADKF_MFMPREC     #define (1<<12) = 0x00001000  hardware/adkbits.h: *35
ADKF_MSBSYNC     #define (1<<9) = 0x00000200  hardware/adkbits.h: *38
ADKF_PRE000NS    #define 0 = 0x00000000  hardware/adkbits.h: *49
ADKF_PRE140NS    #define (ADKF_PRECOMP0) = 0x00002000  hardware/adkbits.h: *50
ADKF_PRE280NS    #define (ADKF_PRECOMP1) = 0x00004000  hardware/adkbits.h: *51
ADKF_PRE560NS    #define (ADKF_PRECOMP0|ADKF_PRECOMP1) = 0x00006000
                 hardware/adkbits.h: *52
ADKF_PRECOMP0    #define (1<<13) = 0x00002000  hardware/adkbits.h: *34
ADKF_PRECOMP1    #define (1<<14) = 0x00004000  hardware/adkbits.h: *33
ADKF_SETCLR      #define (1<<15) = 0x00008000  hardware/adkbits.h: *32
ADKF_UARTBRK     #define (1<<11) = 0x00000800  hardware/adkbits.h: *36
ADKF_USE0P1      #define (1<<4) = 0x00000010  hardware/adkbits.h: *43
ADKF_USE0V1      #define (1<<0) = 0x00000001  hardware/adkbits.h: *47
ADKF_USE1P2      #define (1<<5) = 0x00000020  hardware/adkbits.h: *42
ADKF_USE1V2      #define (1<<1) = 0x00000002  hardware/adkbits.h: *46
ADKF_USE2P3      #define (1<<6) = 0x00000040  hardware/adkbits.h: *41
ADKF_USE2V3      #define (1<<2) = 0x00000004  hardware/adkbits.h: *45
```

Include File Cross Reference Page 6

```
ADKF_USE3PN      #define (1<<7) = 0x00000080  hardware/adkbits.h: *40
ADKF_USE3VN      #define (1<<3) = 0x00000008  hardware/adkbits.h: *44
ADKF_WORDSYNC    #define (1<<10) = 0x00000400  hardware/adkbits.h: *37
ADNB_STARTPROC   #define 0 = 0x00000000  libraries/expansion.h: *18
ADNF_STARTPROC   #define (1L<<0) = 0x00000001  libraries/expansion.h: *19
ADO_CommFileLen  #define (ADO_Dummy + 4) = 0x800007d4  dos/dostags.h: *132
ADO_CommNameLen  #define (ADO_Dummy + 3) = 0x800007d3  dos/dostags.h: *130
ADO_DirLen       #define (ADO_Dummy + 2) = 0x800007d2  dos/dostags.h: *128
ADO_Dummy        #define (TAG_USER + 2000) = 0x800007d0  dos/dostags.h: *115
ADO_FH_Mode      #define (ADO_Dummy + 1) = 0x800007d1  dos/dostags.h: *116
ADO_PromptLen    #define (ADO_Dummy + 5) = 0x800007d5  dos/dostags.h: *134
AFB_68010        #define 0 = 0x00000000  exec/execbase.h: *153
AFB_68020        #define 1 = 0x00000001  exec/execbase.h: *154
AFB_68030        #define 2 = 0x00000002  exec/execbase.h: *155
AFB_68040        #define 3 = 0x00000003  exec/execbase.h: *156
AFB_68881        #define 4 = 0x00000004  exec/execbase.h: *157
AFB_68882        #define 5 = 0x00000005  exec/execbase.h: *158
AFB_DISK         #define 1 = 0x00000001  libraries/diskfont.h: *87
AFB_MEMORY       #define 0 = 0x00000000  libraries/diskfont.h: *85
AFB_SCALED       #define 2 = 0x00000002  libraries/diskfont.h: *89
AFB_TAGGED       #define 16 = 0x00000010  libraries/diskfont.h: *92
AFF_68010        #define (1L<<0) = 0x00000001  exec/execbase.h: *160
AFF_68020        #define (1L<<1) = 0x00000002  exec/execbase.h: *161
AFF_68030        #define (1L<<2) = 0x00000004  exec/execbase.h: *162
AFF_68040        #define (1L<<3) = 0x00000008  exec/execbase.h: *163
AFF_68881        #define (1L<<4) = 0x00000010  exec/execbase.h: *164
AFF_68882        #define (1L<<5) = 0x00000020  exec/execbase.h: *165
AFF_DISK         #define 0x0002 = 0x00000002  libraries/diskfont.h: *88
AFF_MEMORY       #define 0x0001 = 0x00000001  libraries/diskfont.h: *86
AFF_SCALED       #define 0x0004 = 0x00000004  libraries/diskfont.h: *90
AFF_TAGGED       #define 0x10000L = 0x00010000  libraries/diskfont.h: *93
AFbias           char in struct narrator_rb +0x0055 devices/narrator.h: *119
AGNUS            #define  graphics/gfx.h: *22, 23
AG_BadParm       #define 0x00080000 = 0x00080000  exec/alerts.h: *53
AG_CloseDev      #define 0x000A0000 = 0x000a0000  exec/alerts.h: *55
AG_CloseLib      #define 0x00090000 = 0x00090000  exec/alerts.h: *54
AG_IOError       #define 0x00060000 = 0x00060000  exec/alerts.h: *51
AG_MakeLib       #define 0x00020000 = 0x00020000  exec/alerts.h: *47
AG_NoMemory      #define 0x00010000 = 0x00010000  exec/alerts.h: *46
AG_NoSignal      #define 0x00070000 = 0x00070000  exec/alerts.h: *52
AG_OpenDev       #define 0x00040000 = 0x00040000  exec/alerts.h: *49
AG_OpenLib       #define 0x00030000 = 0x00030000  exec/alerts.h: *48
AG_OpenRes       #define 0x00050000 = 0x00050000  exec/alerts.h: *50
AG_ProcCreate    #define 0x000B0000 = 0x000b0000  exec/alerts.h: *56
ALERTLAYERSNOMEM #define 0x83010000 = 0x83010000  graphics/layers.h: *52
ALERT_TYPE       #define 0x80000000 = 0x80000000  intuition/intuition.h: *1310
ALPHA_P_101      #define 0x01 = 0x00000001  intuition/preferences.h: *193
ALTKEYMAP        #define GACT_ALTKEYMAP = 0x00001000
                 intuition/iobsolete.h: *84
ALTLEFT          #define (IEQUALIFIER_LALT) = 0x00000010
                 intuition/intuition.h: *1336
ALTRIGHT         #define (IEQUALIFIER_RALT) = 0x00000020
                 intuition/intuition.h: *1337
AMIGAKEYS        #define (AMIGALEFT | AMIGARIGHT) = 0x000000c0
                 intuition/intuition.h: *1340
AMIGALEFT        #define (IEQUALIFIER_LCOMMAND) = 0x00000040
                 intuition/intuition.h: *1338
AMIGARIGHT       #define (IEQUALIFIER_RCOMMAND) = 0x00000080
                 intuition/intuition.h: *1339
AM_VERSION       #define 1 = 0x00000001  workbench/workbench.h: *124
ANBC             #define 0x20 = 0x00000020  hardware/blit.h: *34
ANBNC            #define 0x10 = 0x00000010  hardware/blit.h: *35
ANFRACSIZE       #define 6 = 0x00000006  graphics/gels.h: *46
ANIMHALF         #define 0x0020 = 0x00000020  graphics/gels.h: *47
ANTI_ALIAS       #define 0x0800 = 0x00000800  intuition/preferences.h: *259
AN_AddSWGadget   #define 0x8401000A = 0x8401000a  exec/alerts.h: *151
```

```
AN_AsyncPkt       #define 0x07000004 = 0x07000004  exec/alerts.h: *166
AN_AudioDev       #define 0x10000000 = 0x10000000  exec/alerts.h: *193
AN_BadChkSum      #define 0x07000009 = 0x07000009  exec/alerts.h: *171
AN_BadExpansionFree #define 0x0A000001 = 0x0a000001  exec/alerts.h: *187
AN_BadFreeAddr    #define 0x0100000F = 0x0100000f  exec/alerts.h: *116
AN_BadGadget      #define 0x04000001 = 0x04000001  exec/alerts.h: *142
AN_BadInitFunc    #define 0x0700000D = 0x0700000d  exec/alerts.h: *175
AN_BadMessage     #define 0x8400000D = 0x8400000d  exec/alerts.h: *154
AN_BadOverlay     #define 0x0700000C = 0x0700000c  exec/alerts.h: *174
AN_BadSegList     #define 0x08000001 = 0x08000001  exec/alerts.h: *180
AN_BadState       #define 0x8400000C = 0x8400000c  exec/alerts.h: *153
AN_BaseChkSum     #define 0x01000002 = 0x01000002  exec/alerts.h: *100
AN_BitMap         #define 0x07000007 = 0x07000007  exec/alerts.h: *169
AN_BltBitMap      #define 0x8201000A = 0x8201000a  exec/alerts.h: *125
AN_BogusExcpt     #define 0x8100000A = 0x8100000a  exec/alerts.h: *109
AN_BootError      #define 0x30000001 = 0x30000001  exec/alerts.h: *228
AN_BootStrap      #define 0x30000000 = 0x30000000  exec/alerts.h: *227
AN_CIARsrc        #define 0x20000000 = 0x20000000  exec/alerts.h: *216
AN_ConsoleDev     #define 0x11000000 = 0x11000000  exec/alerts.h: *196
AN_CreatePort     #define 0x84010002 = 0x84010002  exec/alerts.h: *143
AN_DOSLib         #define 0x07000000 = 0x07000000  exec/alerts.h: *162
AN_DRHasDisk      #define 0x21000001 = 0x21000001  exec/alerts.h: *220
AN_DRIntNoAct     #define 0x21000002 = 0x21000002  exec/alerts.h: *221
AN_DiskBlkSeq     #define 0x07000006 = 0x07000006  exec/alerts.h: *168
AN_DiskCopy       #define 0x32000000 = 0x32000000  exec/alerts.h: *252
AN_DiskError      #define 0x0700000A = 0x0700000a  exec/alerts.h: *172
AN_DiskRsrc       #define 0x21000000 = 0x21000000  exec/alerts.h: *219
AN_DiskfontLib    #define 0x0B000000 = 0x0b000000  exec/alerts.h: *190
AN_EndTask        #define 0x07000002 = 0x07000002  exec/alerts.h: *164
AN_ExcptVect      #define 0x01000001 = 0x01000001  exec/alerts.h: *99
AN_ExecLib        #define 0x01000000 = 0x01000000  exec/alerts.h: *98
AN_ExpansionLib   #define 0x0A000000 = 0x0a000000  exec/alerts.h: *186
AN_FileReclosed   #define 0x0700000E = 0x0700000e  exec/alerts.h: *176
AN_FreeTwice      #define 0x01000009 = 0x01000009  exec/alerts.h: *108
AN_FreeVec        #define 0x07000005 = 0x07000005  exec/alerts.h: *167
AN_GadTools       #define 0x33000000 = 0x33000000  exec/alerts.h: *255
AN_GadgetType     #define 0x84000001 = 0x84000001  exec/alerts.h: *141
AN_GamePortDev    #define 0x12000000 = 0x12000000  exec/alerts.h: *200
AN_GfxFreeError   #define 0x0200000D = 0x0200000d  exec/alerts.h: *129
AN_GfxNewError    #define 0x0200000C = 0x0200000c  exec/alerts.h: *128
AN_GfxNoLCM       #define 0x82011234 = 0x82011234  exec/alerts.h: *131
AN_GfxNoMem       #define 0x82010000 = 0x82010000  exec/alerts.h: *120
AN_GfxNoMemMspc   #define 0x82010001 = 0x82010001  exec/alerts.h: *121
AN_GraphicsLib    #define 0x02000000 = 0x02000000  exec/alerts.h: *119
AN_IOAfterClose   #define 0x0100000D = 0x0100000d  exec/alerts.h: *113
AN_IOUsedTwice    #define 0x0100000B = 0x0100000b  exec/alerts.h: *110
AN_IconLib        #define 0x09000000 = 0x09000000  exec/alerts.h: *183
AN_InitAPtr       #define 0x01000007 = 0x01000007  exec/alerts.h: *105
AN_IntrMem        #define 0x81000006 = 0x81000006  exec/alerts.h: *104
AN_Intuition      #define 0x04000000 = 0x04000000  exec/alerts.h: *140
AN_ItemAlloc      #define 0x04010003 = 0x04010003  exec/alerts.h: *144
AN_ItemBoxTop     #define 0x84000006 = 0x84000006  exec/alerts.h: *147
AN_KeyFree        #define 0x07000008 = 0x07000008  exec/alerts.h: *170
AN_KeyRange       #define 0x0700000B = 0x0700000b  exec/alerts.h: *173
AN_KeyboardDev    #define 0x13000000 = 0x13000000  exec/alerts.h: *203
AN_LayersLib      #define 0x03000000 = 0x03000000  exec/alerts.h: *136
AN_LayersNoMem    #define 0x83010000 = 0x83010000  exec/alerts.h: *137
AN_LibChkSum      #define 0x01000003 = 0x01000003  exec/alerts.h: *101
AN_LongFrame      #define 0x82010006 = 0x82010006  exec/alerts.h: *122
AN_MakeVPort      #define 0x82010030 = 0x82010030  exec/alerts.h: *127
AN_MathLib        #define 0x05000000 = 0x05000000  exec/alerts.h: *159
AN_MemCorrupt     #define 0x81000005 = 0x81000005  exec/alerts.h: *103
AN_MemoryInsane   #define 0x0100000C = 0x0100000c  exec/alerts.h: *112
AN_MiscRsrc       #define 0x22000000 = 0x22000000  exec/alerts.h: *224
AN_NoConsole      #define 0x8400000F = 0x8400000f  exec/alerts.h: *156
AN_NoFonts        #define 0xB1000001 = 0xb1000001  exec/alerts.h: *232
```

```
AN_NoWindow       #define 0x11000001 = 0x11000001  exec/alerts.h: *197
AN_ObsoleteFont   #define 0x02000401 = 0x02000401  exec/alerts.h: *133
AN_OpenScreen     #define 0x84010007 = 0x84010007  exec/alerts.h: *148
AN_OpenScrnRast   #define 0x84010008 = 0x84010008  exec/alerts.h: *149
AN_OpenWindow     #define 0x8401000B = 0x8401000b  exec/alerts.h: *152
AN_PlaneAlloc     #define 0x84010005 = 0x84010005  exec/alerts.h: *146
AN_QPktFail       #define 0x07000003 = 0x07000003  exec/alerts.h: *165
AN_RAMLib         #define 0x08000000 = 0x08000000  exec/alerts.h: *179
AN_RegionMemory   #define 0x8201000B = 0x8201000b  exec/alerts.h: *126
AN_SemCorrupt     #define 0x01000008 = 0x01000008  exec/alerts.h: *107
AN_ShortFrame     #define 0x82010007 = 0x82010007  exec/alerts.h: *123
AN_StackProbe     #define 0x0100000E = 0x0100000e  exec/alerts.h: *114
AN_StartMem       #define 0x07010001 = 0x07010001  exec/alerts.h: *163
AN_SubAlloc       #define 0x04010004 = 0x04010004  exec/alerts.h: *145
AN_SysScrnType    #define 0x84000009 = 0x84000009  exec/alerts.h: *150
AN_TDCalibSeek    #define 0x14000001 = 0x14000001  exec/alerts.h: *207
AN_TDDelay        #define 0x14000002 = 0x14000002  exec/alerts.h: *208
AN_TMBadReq       #define 0x15000001 = 0x15000001  exec/alerts.h: *212
AN_TMBadSupply    #define 0x15000002 = 0x15000002  exec/alerts.h: *213
AN_TextTmpRas     #define 0x02010009 = 0x02010009  exec/alerts.h: *124
AN_TimerDev       #define 0x15000000 = 0x15000000  exec/alerts.h: *211
AN_TrackDiskDev   #define 0x14000000 = 0x14000000  exec/alerts.h: *206
AN_Unknown        #define 0x35000000 = 0x35000000  exec/alerts.h: *261
AN_UtilityLib     #define 0x34000000 = 0x34000000  exec/alerts.h: *258
AN_WBAddToolMenuItem #define 0xB1010008 = 0xb1010008  exec/alerts.h: *241
AN_WBBadIOMsg     #define 0x31000003 = 0x31000003  exec/alerts.h: *235
AN_WBBadStartupMsg1 #define 0x31000001 = 0x31000001  exec/alerts.h: *233
AN_WBBadStartupMsg2 #define 0x31000002 = 0x31000002  exec/alerts.h: *234
AN_WBCreateWBMenusCreateMenus1 #define 0xB1010005 = 0xb1010005
                  exec/alerts.h: *238
AN_WBCreateWBMenusCreateMenus2 #define 0xB1010006 = 0xb1010006
                  exec/alerts.h: *239
AN_WBInitLayerDemon #define 0xB101000B = 0xb101000b  exec/alerts.h: *244
AN_WBInitPotionAllocDrawer #define 0xB1010004 = 0xb1010004
                  exec/alerts.h: *237
AN_WBInitScreenAndWindows1 #define 0xB101000D = 0xb101000d
                  exec/alerts.h: *246
AN_WBInitScreenAndWindows2 #define 0xB101000E = 0xb101000e
                  exec/alerts.h: *247
AN_WBInitScreenAndWindows3 #define 0xB101000F = 0xb101000f
                  exec/alerts.h: *248
AN_WBLayoutWBMenusLayoutMenus #define 0xB1010007 = 0xb1010007
                  exec/alerts.h: *240
AN_WBMAlloc       #define 0xB1010010 = 0xb1010010  exec/alerts.h: *249
AN_WBReLayoutToolMenu #define 0xB1010009 = 0xb1010009  exec/alerts.h: *242
AN_WBinitTimer    #define 0xB101000A = 0xb101000a  exec/alerts.h: *243
AN_WBinitWbGels   #define 0xB101000C = 0xb101000c  exec/alerts.h: *245
AN_WeirdEcho      #define 0x8400000E = 0x8400000e  exec/alerts.h: *155
AN_Workbench      #define 0x31000000 = 0x31000000  exec/alerts.h: *231
AO_AudioDev       #define 0x00008010 = 0x00008010  exec/alerts.h: *72
AO_BootStrap      #define 0x00008030 = 0x00008030  exec/alerts.h: *83
AO_CIARsrc        #define 0x00008020 = 0x00008020  exec/alerts.h: *79
AO_ConsoleDev     #define 0x00008011 = 0x00008011  exec/alerts.h: *73
AO_DOSLib         #define 0x00008007 = 0x00008007  exec/alerts.h: *65
AO_DiskCopy       #define 0x00008032 = 0x00008032  exec/alerts.h: *85
AO_DiskRsrc       #define 0x00008021 = 0x00008021  exec/alerts.h: *80
AO_DiskfontLib    #define 0x0000800B = 0x0000800b  exec/alerts.h: *69
AO_ExecLib        #define 0x00008001 = 0x00008001  exec/alerts.h: *60
AO_ExpansionLib   #define 0x0000800A = 0x0000800a  exec/alerts.h: *68
AO_GadTools       #define 0x00008033 = 0x00008033  exec/alerts.h: *86
AO_GamePortDev    #define 0x00008012 = 0x00008012  exec/alerts.h: *74
AO_GraphicsLib    #define 0x00008002 = 0x00008002  exec/alerts.h: *61
AO_IconLib        #define 0x00008009 = 0x00008009  exec/alerts.h: *67
AO_Intuition      #define 0x00008004 = 0x00008004  exec/alerts.h: *63
AO_KeyboardDev    #define 0x00008013 = 0x00008013  exec/alerts.h: *75
AO_LayersLib      #define 0x00008003 = 0x00008003  exec/alerts.h: *62
```

Include File Cross Reference Page 9

```
AO_MathLib        #define 0x00008005 = 0x00008005  exec/alerts.h: *64
AO_MiscRsrc       #define 0x00008022 = 0x00008022  exec/alerts.h: *81
AO_RAMLib         #define 0x00008008 = 0x00008008  exec/alerts.h: *66
AO_TimerDev       #define 0x00008015 = 0x00008015  exec/alerts.h: *77
AO_TrackDiskDev   #define 0x00008014 = 0x00008014  exec/alerts.h: *76
AO_Unknown        #define 0x00008035 = 0x00008035  exec/alerts.h: *87
AO_UtilityLib     #define 0x0000800C = 0x0000800c  exec/alerts.h: *70
AO_Workbench      #define 0x00008031 = 0x00008031  exec/alerts.h: *84
AOlPen            char in struct RastPort +0x001b graphics/rastport.h: *67
APB_DIDDIR        #define 3 = 0x00000003  dos/dosasl.h: *85
APB_DODIR         #define 2 = 0x00000002  dos/dosasl.h: *80
APB_DODOT         #define 5 = 0x00000005  dos/dosasl.h: *91
APB_DOWILD        #define 0 = 0x00000000  dos/dosasl.h: *71
APB_DirChanged    #define 6 = 0x00000006  dos/dosasl.h: *94
APB_ITSWILD       #define 1 = 0x00000001  dos/dosasl.h: *74
APB_NOMEMERR      #define 4 = 0x00000004  dos/dosasl.h: *88
APF_DIDDIR        #define 8 = 0x00000008  dos/dosasl.h: *86
APF_DODIR         #define 4 = 0x00000004  dos/dosasl.h: *81
APF_DODOT         #define 32 = 0x00000020  dos/dosasl.h: *92
APF_DOWILD        #define 1 = 0x00000001  dos/dosasl.h: *72
APF_DirChanged    #define 64 = 0x00000040  dos/dosasl.h: *95
APF_ITSWILD       #define 2 = 0x00000002  dos/dosasl.h: *75
APF_NOMEMERR      #define 16 = 0x00000010  dos/dosasl.h: *89
APTR              typedef pointer to void  exec/types.h: *36
APTR_TYPEDEF      #define  exec/types.h: *35
AREAOUTLINE       #define 0x08 = 0x00000008  graphics/rastport.h: *107
ARG0              macro (1 argument)   rexx/storage.h: *117
ARG1              macro (1 argument)   rexx/storage.h: *118
ARG2              macro (1 argument)   rexx/storage.h: *119
ARROWIDCMP        #define (IDCMP_GADGETUP | IDCMP_GADGETDOWN | IDCMP_INTUITICKS
                  | IDCMP_MOUSEBUTTONS) = 0x00400068
                  libraries/gadtools.h: *64
ASHIFTSHIFT       #define 12 = 0x0000000c  hardware/blit.h: *63
ASL_BackPen       #define ASL_Dummy+15 = 0x8008000f  libraries/asl.h: *210
ASL_CancelText    #define ASL_Dummy+19 = 0x80080013  libraries/asl.h: *215
ASL_Dir           #define ASL_Dummy+9 = 0x80080009  libraries/asl.h: *202
ASL_Dummy         #define (TAG_USER + 0x80000) = 0x80080000
                  libraries/asl.h: *190
ASL_ExtFlags1     #define ASL_Dummy+22 = 0x80080016  libraries/asl.h: *219
ASL_File          #define ASL_Dummy+8 = 0x80080008  libraries/asl.h: *201
ASL_FileRequest   #define 0 = 0x00000000  libraries/asl.h: *183
ASL_FontFlags     #define ASL_Dummy+13 = 0x8008000d  libraries/asl.h: *208
ASL_FontHeight    #define ASL_Dummy+11 = 0x8008000b  libraries/asl.h: *206
ASL_FontName      #define ASL_Dummy+10 = 0x8008000a  libraries/asl.h: *205
ASL_FontRequest   #define 1 = 0x00000001  libraries/asl.h: *184
ASL_FontStyles    #define ASL_Dummy+12 = 0x8008000c  libraries/asl.h: *207
ASL_FrontPen      #define ASL_Dummy+14 = 0x8008000e  libraries/asl.h: *209
ASL_FuncFlags     #define ASL_Dummy+20 = 0x80080014  libraries/asl.h: *216
ASL_Hail          #define ASL_Dummy+1 = 0x80080001  libraries/asl.h: *192
ASL_Height        #define ASL_Dummy+6 = 0x80080006  libraries/asl.h: *197
ASL_HookFunc      #define ASL_Dummy+7 = 0x80080007  libraries/asl.h: *198
ASL_LeftEdge      #define ASL_Dummy+3 = 0x80080003  libraries/asl.h: *194
ASL_MaxHeight     #define ASL_Dummy+17 = 0x80080011  libraries/asl.h: *212
ASL_MinHeight     #define ASL_Dummy+16 = 0x80080010  libraries/asl.h: *211
ASL_ModeList      #define ASL_Dummy+21 = 0x80080015  libraries/asl.h: *218
ASL_OKText        #define ASL_Dummy+18 = 0x80080012  libraries/asl.h: *214
ASL_Pattern       #define ASL_FontName = 0x8008000a  libraries/asl.h: *221
ASL_TopEdge       #define ASL_Dummy+4 = 0x80080004  libraries/asl.h: *195
ASL_Width         #define ASL_Dummy+5 = 0x80080005  libraries/asl.h: *196
ASL_Window        #define ASL_Dummy+2 = 0x80080002  libraries/asl.h: *193
ASPECT_HORIZ      #define 0x00 = 0x00000000  intuition/preferences.h: *176
ASPECT_VERT       #define 0x01 = 0x00000001  intuition/preferences.h: *177
AT_DeadEnd        #define 0x80000000 = 0x80000000  exec/alerts.h: *41
AT_Recovery       #define 0x00000000 = 0x00000000  exec/alerts.h: *42
AUDIONAME         #define "audio.device"  devices/audio.h: *19
AUL               #define 0x4 = 0x00000004  hardware/blit.h: *78
```

Include File Cross Reference Page 10

```
AUTOBACKPEN       #define 1 = 0x00000001  intuition/intuition.h: *1321
AUTODRAWMODE      #define JAM2 = 0x00000001  intuition/intuition.h: *1322
AUTOFRONTPEN      #define 0 = 0x00000000  intuition/intuition.h: *1320
AUTOITEXTFONT     #define NULL = 0x00000000  intuition/intuition.h: *1325
AUTOKNOB          #define 0x0001 = 0x00000001  intuition/intuition.h: *497
AUTOLEFTEDGE      #define 6 = 0x00000006  intuition/intuition.h: *1323
AUTONEXTTEXT      #define NULL = 0x00000000  intuition/intuition.h: *1326
AUTOSCROLL        #define 0x4000 = 0x00004000  intuition/screens.h: *182
AUTOTOPEDGE       #define 3 = 0x00000003  intuition/intuition.h: *1324
AUserExt          short int in struct AnimOb +0x0028 graphics/gels.h: *229
AUserStuff        #define WORD  graphics/gels.h: *66, 229
AVbias            char in struct narrator_rb +0x0054 devices/narrator.h: *118
A_OR_B            #define ABC|ANBC|NABC | ABNC|ANBNC|NABNC = 0x000000fc
                  hardware/blit.h: *42
A_OR_C            #define ABC|NABC|ABNC | ANBC|NANBC|ANBNC = 0x000000fa
                  hardware/blit.h: *43
A_TO_D            #define ABC|ANBC|ABNC|ANBNC = 0x000000f0
                  hardware/blit.h: *45
A_XOR_C           #define NABC|ABNC | NANBC|ANBNC = 0x0000005a
                  hardware/blit.h: *44
ActiView          pointer to struct View in struct GfxBase
         +0x0022 graphics/gfxbase.h: *28
ActiViewCprSemaphore pointer to struct SignalSemaphore in struct GfxBase
         +0x019a graphics/gfxbase.h: *90
Actions           unsigned long int in struct SGWork
         +0x001e intuition/sghooks.h: *46
Activation        unsigned short int in struct Gadget
         +0x000e intuition/intuition.h: *225
ActivePens        array [2] of unsigned char in struct StringExtend
         +0x0006 intuition/sghooks.h: *23
ActiveScreen      pointer to struct Screen in struct IntuitionBase
         +0x0038 intuition/intuitionbase.h: *75
ActiveWindow      pointer to struct Window in struct IntuitionBase
         +0x0034 intuition/intuitionbase.h: *74
After             pointer to struct Bob in struct Bob
         +0x000e graphics/gels.h: *159
AlertData         pointer to void in struct ExecBase
         +0x004a exec/execbase.h: *52
AlgoStyle         unsigned char in struct RastPort
         +0x0038 graphics/rastport.h: *79
AltKeyMap         pointer to struct KeyMap in struct StringInfo
         +0x0020 intuition/intuition.h: *560
AnOldX            short int in struct AnimOb +0x000e graphics/gels.h: *213
AnOldY            short int in struct AnimOb +0x000c graphics/gels.h: *213
AnX               short int in struct AnimOb +0x0012 graphics/gels.h: *216
AnY               short int in struct AnimOb +0x0010 graphics/gels.h: *216
AnalogSignalInterval structure tag
     size 0x0004 graphics/monitor.h: *136, 150, 151, 152, 153
AnchorPath        structure tag size 0x0119 dos/dosasl.h: *54
AnimBob           pointer to struct Bob in struct AnimComp
         +0x0022 graphics/gels.h: *202
AnimCRoutine      pointer to function returning short int in struct AnimComp
         +0x0016 graphics/gels.h: *195
AnimComp          structure tag
     size 0x0026 graphics/gels.h: *163, 170, 188, 189, 192, 193,
                                227
AnimORoutine      pointer to function returning short int in struct AnimOb
         +0x0020 graphics/gels.h: *224
AnimOb            structure tag size 0x002a graphics/gels.h: *200, 205, 208
AppIcon           structure tag size 0x0004 workbench/workbench.h: *147
AppMenuItem       structure tag size 0x0004 workbench/workbench.h: *148
AppMessage        structure tag size 0x0056 workbench/workbench.h: *126
AppWindow         structure tag size 0x0004 workbench/workbench.h: *146
AreaCircle        macro (4 arguments)   graphics/gfxmacros.h: *44
AreaInfo          structure tag size 0x0018 graphics/rastport.h: *23, 62
AreaInfo          pointer to struct AreaInfo in struct RastPort
```

```
         +0x0010 graphics/rastport.h: *62
AreaPtSz         char in struct RastPort +0x001d graphics/rastport.h: *69
AreaPtrn         pointer to unsigned short int in struct RastPort
         +0x0008 graphics/rastport.h: *60
AslName          #define "asl.library"  libraries/asl.h: *50
AssignList       structure tag size 0x0008 dos/dosextens.h: *386, 396, 397
AttnFlags        unsigned short int in struct ExecBase
         +0x0128 exec/execbase.h: *73
AttnResched      unsigned short int in struct ExecBase
         +0x012a exec/execbase.h: *75
AudChannel       structure tag (size 0x0010 bytes)in struct Custom
                 hardware/custom.h: *97
AvailFonts       structure tag size 0x000a libraries/diskfont.h: *95
AvailFontsHeader structure tag size 0x0002 libraries/diskfont.h: *105
back             pointer to struct Layer in struct Layer
         +0x0004 graphics/clip.h: *36
backgroundPen    #define BACKGROUNDPEN = 0x00000007
                 intuition/iobsolete.h: *268
bb_chksum        long int in struct BootBlock +0x0004 devices/bootblock.h: *21
bb_dosblock      long int in struct BootBlock +0x0008 devices/bootblock.h: *22
bb_id            array [4] of unsigned char in struct BootBlock
         +0x0000 devices/bootblock.h: *20
bbb_BlockPairs   array [61] of struct BadBlockEntry(size 0x0008 bytes) in
                 struct BadBlockBlock
         +0x0018 devices/hardblocks.h: *125
bbb_ChkSum       long int in struct BadBlockBlock
         +0x0008 devices/hardblocks.h: *121
bbb_HostID       unsigned long int in struct BadBlockBlock
         +0x000c devices/hardblocks.h: *122
bbb_ID           unsigned long int in struct BadBlockBlock
         +0x0000 devices/hardblocks.h: *119
bbb_Next         unsigned long int in struct BadBlockBlock
         +0x0010 devices/hardblocks.h: *123
bbb_Reserved     unsigned long int in struct BadBlockBlock
         +0x0014 devices/hardblocks.h: *124
bbb_SummedLongs unsigned long int in struct BadBlockBlock
         +0x0004 devices/hardblocks.h: *120
bbe_BadBlock     unsigned long int in struct BadBlockEntry
         +0x0000 devices/hardblocks.h: *114
bbe_GoodBlock    unsigned long int in struct BadBlockEntry
         +0x0004 devices/hardblocks.h: *115
beamcon0         unsigned short int in struct Custom
         +0x01dc hardware/custom.h: *137
beamsync         short int in struct bltnode +0x000c hardware/blit.h: *96
blitbuff         pointer to short int in struct Layer_Info
         +0x005e graphics/layers.h: *47
blitsize         short int in struct bltnode +0x000a hardware/blit.h: *95
blitter          pointer to long int in struct GfxBase
         +0x002e graphics/gfxbase.h: *31
blockPen         #define BLOCKPEN = 0x00000001  intuition/iobsolete.h: *262
bltadat          unsigned short int in struct Custom
         +0x0074 hardware/custom.h: *79
bltafwm          unsigned short int in struct Custom
         +0x0044 hardware/custom.h: *61
bltalwm          unsigned short int in struct Custom
         +0x0046 hardware/custom.h: *62
bltamod          unsigned short int in struct Custom
         +0x0064 hardware/custom.h: *74
bltapt           pointer to void in struct Custom
         +0x0050 hardware/custom.h: *65
bltbdat          unsigned short int in struct Custom
         +0x0072 hardware/custom.h: *78
bltbmod          unsigned short int in struct Custom
         +0x0062 hardware/custom.h: *73
bltbpt           pointer to void in struct Custom
         +0x004c hardware/custom.h: *64
```

```
bltcdat          unsigned short int in struct Custom
         +0x0070 hardware/custom.h: *77
bltcmod          unsigned short int in struct Custom
         +0x0060 hardware/custom.h: *72
bltcon0          unsigned short int in struct Custom
         +0x0040 hardware/custom.h: *59
bltcon0l         unsigned char in struct Custom +0x005b hardware/custom.h: *69
bltcon1          unsigned short int in struct Custom
         +0x0042 hardware/custom.h: *60
bltcpt           pointer to void in struct Custom
         +0x0048 hardware/custom.h: *63
bltddat          unsigned short int in struct Custom
         +0x0000 hardware/custom.h: *28
bltdmod          unsigned short int in struct Custom
         +0x0066 hardware/custom.h: *75
bltdpt           pointer to void in struct Custom
         +0x0054 hardware/custom.h: *66
blthd            pointer to struct bltnode in struct GfxBase
         +0x003a graphics/gfxbase.h: *34
bltnode          structure tag size 0x0012 graphics/gfxbase.h: *34, 35
                 hardware/blit.h: 90, 92
bltsize          unsigned short int in struct Custom
         +0x0058 hardware/custom.h: *67
bltsizh          unsigned short int in struct Custom
         +0x005e hardware/custom.h: *71
bltsizv          unsigned short int in struct Custom
         +0x005c hardware/custom.h: *70
bltsrv           struct Interrupt(size 0x0016 bytes) in struct GfxBase
         +0x0076 graphics/gfxbase.h: *36
bltt1            pointer to struct bltnode in struct GfxBase
         +0x003e graphics/gfxbase.h: *34
bn_DeviceNode    pointer to void in struct BootNode
         +0x0010 libraries/expansionbase.h: *39
bn_Flags         unsigned short int in struct BootNode
         +0x000e libraries/expansionbase.h: *38
bn_Node          struct Node(size 0x000e bytes) in struct BootNode
         +0x0000 libraries/expansionbase.h: *37
bottommost       short int in struct GelsInfo +0x001c graphics/rastport.h: *52
bounds           struct Rectangle(size 0x0008 bytes) in struct Layer
         +0x0010 graphics/clip.h: *39
bounds           struct Rectangle(size 0x0008 bytes) in struct ClipRect
         +0x0010 graphics/clip.h: *70
bounds           struct Rectangle(size 0x0008 bytes) in struct RegionRectangle
         +0x0008 graphics/regions.h: *26
bounds           struct Rectangle(size 0x0008 bytes) in struct Region
         +0x0000 graphics/regions.h: *31
bpl1mod          unsigned short int in struct Custom
         +0x0108 hardware/custom.h: *110
bpl2mod          unsigned short int in struct Custom
         +0x010a hardware/custom.h: *111
bplcon0          unsigned short int in struct Custom
         +0x0100 hardware/custom.h: *106
bplcon1          unsigned short int in struct Custom
         +0x0102 hardware/custom.h: *107
bplcon2          unsigned short int in struct Custom
         +0x0104 hardware/custom.h: *108
bplcon3          unsigned short int in struct Custom
         +0x0106 hardware/custom.h: *109
bpldat           array [8] of unsigned short int in struct Custom
         +0x0110 hardware/custom.h: *114
bplhmod          unsigned short int in struct Custom
         +0x010c hardware/custom.h: *112
bplhstop         unsigned short int in struct Custom
         +0x01d6 hardware/custom.h: *134
bplhstrt         unsigned short int in struct Custom
         +0x01d4 hardware/custom.h: *133
```

Include File Cross Reference Page 13

```
bplpt           array [8] of pointer to void in struct Custom
        +0x00e0 hardware/custom.h: *105
bsa_DestBitMap  pointer to struct BitMap in struct BitScaleArgs
        +0x001c graphics/scale.h: *27
bsa_DestHeight  unsigned short int in struct BitScaleArgs
        +0x0012 graphics/scale.h: *24
bsa_DestWidth   unsigned short int in struct BitScaleArgs
        +0x0010 graphics/scale.h: *24
bsa_DestX       unsigned short int in struct BitScaleArgs
        +0x000c graphics/scale.h: *23
bsa_DestY       unsigned short int in struct BitScaleArgs
        +0x000e graphics/scale.h: *23
bsa_Flags       unsigned long int in struct BitScaleArgs
        +0x0020 graphics/scale.h: *28
bsa_Reserved1   long int in struct BitScaleArgs +0x0028 graphics/scale.h: *30
bsa_Reserved2   long int in struct BitScaleArgs +0x002c graphics/scale.h: *31
bsa_SrcBitMap   pointer to struct BitMap in struct BitScaleArgs
        +0x0018 graphics/scale.h: *26
bsa_SrcHeight   unsigned short int in struct BitScaleArgs
        +0x0006 graphics/scale.h: *21
bsa_SrcWidth    unsigned short int in struct BitScaleArgs
        +0x0004 graphics/scale.h: *21
bsa_SrcX        unsigned short int in struct BitScaleArgs
        +0x0000 graphics/scale.h: *20
bsa_SrcY        unsigned short int in struct BitScaleArgs
        +0x0002 graphics/scale.h: *20
bsa_XDDA        unsigned short int in struct BitScaleArgs
        +0x0024 graphics/scale.h: *29
bsa_XDestFactor unsigned short int in struct BitScaleArgs
        +0x0014 graphics/scale.h: *25
bsa_XSrcFactor  unsigned short int in struct BitScaleArgs
        +0x0008 graphics/scale.h: *22
bsa_YDDA        unsigned short int in struct BitScaleArgs
        +0x0026 graphics/scale.h: *29
bsa_YDestFactor unsigned short int in struct BitScaleArgs
        +0x0016 graphics/scale.h: *25
bsa_YSrcFactor  unsigned short int in struct BitScaleArgs
        +0x000a graphics/scale.h: *22
bsblthd         pointer to struct bltnode in struct GfxBase
        +0x0042 graphics/gfxbase.h: *35
bsbltt1         pointer to struct bltnode in struct GfxBase
        +0x0046 graphics/gfxbase.h: *35
bytereserved    unsigned char in struct GfxBase
        +0x00a7 graphics/gfxbase.h: *45
B2BOBBER        #define 2 = 0x00000002  graphics/gels.h: *260
B2NORM          #define 0 = 0x00000000  graphics/gels.h: *258
B2SWAP          #define 1 = 0x00000001  graphics/gels.h: *259
BACKDROP        #define WFLG_BACKDROP = 0x00000100
                intuition/iobsolete.h: *156
BACKGROUNDPEN   #define (0x0007) = 0x00000007  intuition/screens.h: *89
BACKSAVED       #define 0x0100 = 0x00000100  graphics/gels.h: *27
BADDR           macro (1 argument)   dos/dos.h: *110
BASEOBJECT      macro (1 argument)   intuition/classes.h: *77
BATTCLOCKNAME   #define "battclock.resource"  resources/battclock.h: *15
BATTMEMNAME     #define "battmem.resource"  resources/battmem.h: *15
BATTMEM_AMIGA_AMNESIA_ADDR #define 0 = 0x00000000
                resources/battmembitsamiga.h: *32
BATTMEM_AMIGA_AMNESIA_LEN #define 1 = 0x00000001
                resources/battmembitsamiga.h: *33
BATTMEM_SCSI_HOST_ID_ADDR #define 65 = 0x00000041
                resources/battmembitsshared.h: *44
BATTMEM_SCSI_HOST_ID_LEN #define 3 = 0x00000003
                resources/battmembitsshared.h: *45
BATTMEM_SCSI_LUNS_ADDR #define 2 = 0x00000002
                resources/battmembitsamiga.h: *61
BATTMEM_SCSI_LUNS_LEN #define 1 = 0x00000001
```

Include File Cross Reference Page 14

```
                resources/battmembitsamiga.h: *62
BATTMEM_SCSI_SYNC_XFER_ADDR #define 68 = 0x00000044
                resources/battmembitsshared.h: *58
BATTMEM_SCSI_SYNC_XFER_LEN #define 1 = 0x00000001
                resources/battmembitsshared.h: *59
BATTMEM_SCSI_TIMEOUT_ADDR #define 1 = 0x00000001
                resources/battmembitsamiga.h: *47
BATTMEM_SCSI_TIMEOUT_LEN #define 1 = 0x00000001
                resources/battmembitsamiga.h: *48
BATTMEM_SHARED_AMNESIA_ADDR #define 64 = 0x00000040
                resources/battmembitsshared.h: *32
BATTMEM_SHARED_AMNESIA_LEN #define 1 = 0x00000001
                resources/battmembitsshared.h: *33
BAUD_110        #define 0x00 = 0x00000000  intuition/preferences.h: *145
BAUD_1200       #define 0x02 = 0x00000002  intuition/preferences.h: *147
BAUD_19200      #define 0x06 = 0x00000006  intuition/preferences.h: *151
BAUD_2400       #define 0x03 = 0x00000003  intuition/preferences.h: *148
BAUD_300        #define 0x01 = 0x00000001  intuition/preferences.h: *146
BAUD_4800       #define 0x04 = 0x00000004  intuition/preferences.h: *149
BAUD_9600       #define 0x05 = 0x00000005  intuition/preferences.h: *150
BAUD_MIDI       #define 0x07 = 0x00000007  intuition/preferences.h: *152
BBID_DOS        #define { 'D', 'O', 'S', '\0' }  devices/bootblock.h: *27
BBID_KICK       #define { 'K', 'I', 'C', 'K' }  devices/bootblock.h: *28
BBNAME_DOS      #define 0x444F5300 = 0x444f5300  devices/bootblock.h: *30
BBNAME_KICK     #define 0x4B49434B = 0x4b49434b  devices/bootblock.h: *31
BCOB_DEST       #define 8 = 0x00000008  hardware/blit.h: *47
BCOB_SRCA       #define 11 = 0x0000000b  hardware/blit.h: *50
BCOB_SRCB       #define 10 = 0x0000000a  hardware/blit.h: *49
BCOB_SRCC       #define 9 = 0x00000009  hardware/blit.h: *48
BCOF_DEST       #define 0x100 = 0x00000100  hardware/blit.h: *51
BCOF_SRCA       #define 0x800 = 0x00000800  hardware/blit.h: *54
BCOF_SRCB       #define 0x400 = 0x00000400  hardware/blit.h: *53
BCOF_SRCC       #define 0x200 = 0x00000200  hardware/blit.h: *52
BC1F_DESC       #define 2 = 0x00000002  hardware/blit.h: *56
BDRAWN          #define 0x0200 = 0x00000200  graphics/gels.h: *39
BEEPING         #define 0x0020 = 0x00000020  intuition/screens.h: *167
BITCLR          #define 0 = 0x00000000  graphics/gfx.h: *20
BITSET          #define 0x8000 = 0x00008000  graphics/gfx.h: *19
BITSPERBYTE     #define 8 = 0x00000008  dos/dos.h: *42
BITSPERLONG     #define 32 = 0x00000020  dos/dos.h: *44
BLITMSG_FAULT   #define 4 = 0x00000004  graphics/gfxbase.h: *100
BLITREVERSE     #define 0x2 = 0x00000002  hardware/blit.h: *74
BLOCKPEN        #define (0x0001) = 0x00000001  intuition/screens.h: *83
BNDRYOFF        macro (1 argument)   graphics/gfxmacros.h: *36
BOBISCOMP       #define 0x0002 = 0x00000002  graphics/gels.h: *36
BOBNIX          #define 0x0800 = 0x00000800  graphics/gels.h: *41
BOBSAWAY        #define 0x0400 = 0x00000400  graphics/gels.h: *40
BOBUPDATE       #define 0x0200 = 0x00000200  graphics/gels.h: *28
BOOL            typedef short  exec/types.h: *66
BOOLEXTEND      #define GACT_BOOLEXTEND = 0x00002000
                intuition/iobsolete.h: *79
BOOLGADGET      #define GTYP_BOOLGADGET = 0x00000001
                intuition/iobsolete.h: *104
BOOLMASK        #define 0x0001 = 0x00000001  intuition/intuition.h: *444
BOOTSECTS       #define 2 = 0x00000002  devices/bootblock.h: *25
BORDERHIT       #define 0 = 0x00000000  graphics/collide.h: *23
BORDERLESS      #define WFLG_BORDERLESS = 0x00000800
                intuition/iobsolete.h: *159
BORDERSNIFF     #define GACT_BORDERSNIFF = 0x00008000
                intuition/iobsolete.h: *77
BORDER_BLANKING #define 0x04 = 0x00000004  graphics/view.h: *142
BORDER_NOTRANSPARENCY #define 0x08 = 0x00000008  graphics/view.h: *143
BOTTOMBORDER    #define GACT_BOTTOMBORDER = 0x00000080
                intuition/iobsolete.h: *76
BOTTOMHIT       #define 2 = 0x00000002  graphics/collide.h: *33
BOUNDED_DIMENSIONS #define 0x0010 = 0x00000010  intuition/preferences.h: *248
```

```
BPLCON2_ZDBPEN    #define (1<<11) = 0x00000800  hardware/custom.h: *166
BPLCON2_ZDBPSEL0  #define (1<<12) = 0x00001000  hardware/custom.h: *167
BPLCON2_ZDBPSEL1  #define (1<<13) = 0x00002000  hardware/custom.h: *168
BPLCON2_ZDBPSEL2  #define (1<<14) = 0x00004000  hardware/custom.h: *169
BPLCON2_ZDCTEN    #define (1<<10) = 0x00000400  hardware/custom.h: *165
BPLCON3_BRDNBLNK  #define (1<<5) = 0x00000020  hardware/custom.h: *176
BPLCON3_BRDNTRAN  #define (1<<4) = 0x00000010  hardware/custom.h: *175
BPLCON3_EXTBLKZD  #define (1<<1) = 0x00000002  hardware/custom.h: *173
BPLCON3_EXTBLNKEN #define (1<<0) = 0x00000001  hardware/custom.h: *172
BPLCON3_ZDCLKEN   #define (1<<2) = 0x00000004  hardware/custom.h: *174
BPTR              typedef long  dos/dos.h: *102
BROADCAST_BEAMCON #define ( LOLDIS | CSBLANK ) = 0x00000808
                  graphics/monitor.h: *132
BROADCAST_HBSTOP  #define 0x27 = 0x00000027  graphics/monitor.h: *127
BROADCAST_HBSTRT  #define 0x01 = 0x00000001  graphics/monitor.h: *124
BROADCAST_HSSTOP  #define 0x17 = 0x00000017  graphics/monitor.h: *126
BROADCAST_HSSTRT  #define 0x06 = 0x00000006  graphics/monitor.h: *125
BROADCAST_VBSTOP  #define 0x1C40 = 0x00001c40  graphics/monitor.h: *131
BROADCAST_VBSTRT  #define 0x0000 = 0x00000000  graphics/monitor.h: *128
BROADCAST_VSSTOP  #define 0x054C = 0x0000054c  graphics/monitor.h: *130
BROADCAST_VSSTRT  #define 0x02A6 = 0x000002a6  graphics/monitor.h: *129
BROTHER_15XL      #define 0x02 = 0x00000002  intuition/preferences.h: *194
BSHIFTSHIFT       #define 12 = 0x0000000c  hardware/blit.h: *64
BSTR              typedef long  dos/dos.h: *103
BUF_FULL          #define 1 = 0x00000001  dos/stdio.h: *28
BUF_LINE          #define 0 = 0x00000000  dos/stdio.h: *27
BUF_NONE          #define 2 = 0x00000002  dos/stdio.h: *29
BUSERFLAGS        #define 0x00FF = 0x000000ff  graphics/gels.h: *34
BUTTONGCLASS      #define "buttongclass"  intuition/classusr.h: *50
BUTTONIDCMP       #define (IDCMP_GADGETUP) = 0x00000040
                  libraries/gadtools.h: *66
BUTTON_KIND       #define 1 = 0x00000001  libraries/gadtools.h: *35
BUserExt          short int in struct Bob +0x001e graphics/gels.h: *167
BUserStuff        #define WORD  graphics/gels.h: *62, 167
BWAITING          #define 0x0100 = 0x00000100  graphics/gels.h: *38
BYTE              char  exec/types.h: *45
BYTEBITS          unsigned char  exec/types.h: *50
BYTEMASK          #define 0xFF = 0x000000ff  exec/types.h: *80
BYTESPERLONG      #define 4 = 0x00000004  dos/dos.h: *43
BackFill          pointer to struct Hook in struct Layer
          +0x0076 graphics/clip.h: *53
BackFill          unsigned char in struct Requester
          +0x001e intuition/intuition.h: *159
BackPen           unsigned char in struct IntuiText
          +0x0001 intuition/intuition.h: *572
BackPen           unsigned char in struct Border
          +0x0005 intuition/intuition.h: *601
BadBlockBlock     structure tag size 0x0200 devices/hardblocks.h: *118
BadBlockEntry     structure tag size 0x0008 devices/hardblocks.h: *113, 125
BarHBorder        char in struct Screen +0x0020 intuition/screens.h: *121
BarHeight         char in struct Screen +0x001e intuition/screens.h: *121
BarLayer          pointer to struct Layer in struct Screen
          +0x014e intuition/screens.h: *145
BarVBorder        char in struct Screen +0x001f intuition/screens.h: *121
BaudRate          unsigned short int in struct Preferences
          +0x0002 intuition/preferences.h: *53
BeamCon0          unsigned short int in struct MonitorSpec
          +0x0028 graphics/monitor.h: *36
BeamSync          short int in struct GfxBase +0x00a2 graphics/gfxbase.h: *42
BeatX             short int in struct Menu +0x001a intuition/intuition.h: *72
BeatY             short int in struct Menu +0x001c intuition/intuition.h: *72
Before            pointer to struct Bob in struct Bob
          +0x000a graphics/gels.h: *158
BgPen             char in struct RastPort +0x001a graphics/rastport.h: *66
BitMap            structure tag size 0x0028 graphics/gfx.h: *48
                  graphics/clip.h: 43, 69
```

```
                  graphics/view.h: 110
                  graphics/rastport.h: 59
                  intuition/intuition.h: 171, 1011, 1058
                  intuition/screens.h: 129, 332, 355
                  graphics/scale.h: 26, 27
BitMap            pointer to struct BitMap in struct ClipRect
          +0x000c graphics/clip.h: *69
BitMap            pointer to struct BitMap in struct RasInfo
          +0x0004 graphics/view.h: *110
BitMap            pointer to struct BitMap in struct RastPort
          +0x0004 graphics/rastport.h: *59
BitMap            pointer to struct BitMap in struct NewWindow
          +0x0022 intuition/intuition.h: *1011
BitMap            pointer to struct BitMap in struct ExtNewWindow
          +0x0022 intuition/intuition.h: *1058
BitMap            struct BitMap(size 0x0028 bytes) in struct Screen
          +0x00b8 intuition/screens.h: *129
BitScaleArgs      structure tag size 0x0030 graphics/scale.h: *19
BlitLock          short int in struct GfxBase +0x00aa graphics/gfxbase.h: *47
BlitNest          short int in struct GfxBase +0x00ac graphics/gfxbase.h: *48
BlitOwner         pointer to struct Task in struct GfxBase
          +0x00bc graphics/gfxbase.h: *51
BlitWaitQ         struct List(size 0x000e bytes) in struct GfxBase
          +0x00ae graphics/gfxbase.h: *50
BlockPen          unsigned char in struct Window
          +0x0063 intuition/intuition.h: *859
BlockPen          unsigned char in struct NewWindow
          +0x0009 intuition/intuition.h: *979
BlockPen          unsigned char in struct ExtNewWindow
          +0x0009 intuition/intuition.h: *1049
BlockPen          unsigned char in struct Screen
          +0x014b intuition/screens.h: *137
BlockPen          unsigned char in struct NewScreen
          +0x000b intuition/screens.h: *314
BlockPen          unsigned char in struct ExtNewScreen
          +0x000b intuition/screens.h: *349
BlockPen          unsigned char in struct (no tag)
          +0x0001 intuition/cghooks.h: *54
Blue              unsigned short int in struct ColorSpec
          +0x0006 intuition/intuition.h: *1245
Bob               structure tag
      size 0x0020 graphics/gels.h: *121, 141, 158, 159, 202
BobComp           pointer to struct AnimComp in struct Bob
          +0x0016 graphics/gels.h: *163
BobVSprite        pointer to struct VSprite in struct Bob
          +0x0012 graphics/gels.h: *161
BoolInfo          structure tag size 0x000a intuition/intuition.h: *428
BootBlock         structure tag size 0x000c devices/bootblock.h: *19
BootNode          structure tag size 0x0014 libraries/expansionbase.h: *35
Border            structure tag
      size 0x0010 intuition/intuition.h: *154, 598, 605
BorderBottom      char in struct Window +0x0039 intuition/intuition.h: *832
BorderLeft        char in struct Window +0x0036 intuition/intuition.h: *832
BorderLine        pointer to short int in struct VSprite
          +0x0028 graphics/gels.h: *115
BorderRPort       pointer to struct RastPort in struct Window
          +0x003a intuition/intuition.h: *833
BorderRight       char in struct Window +0x0038 intuition/intuition.h: *832
BorderTop         char in struct Window +0x0037 intuition/intuition.h: *832
BufBuffer         pointer to short int in struct DBufPacket
          +0x0008 graphics/gels.h: *242
BufPath           pointer to struct VSprite in struct DBufPacket
          +0x0004 graphics/gels.h: *238
BufX              short int in struct DBufPacket +0x0002 graphics/gels.h: *237
BufY              short int in struct DBufPacket +0x0000 graphics/gels.h: *237
Buffer            pointer to unsigned char in struct StringInfo
```

Include File Cross Reference Page 17

```
         +0x0000 intuition/intuition.h: *527
BufferPos        short int in struct StringInfo
         +0x0008 intuition/intuition.h: *529
BufferPos        short int in struct SGWork +0x001a intuition/sghooks.h: *44
BytesPerRow      unsigned short int in struct BitMap
         +0x0000 graphics/gfx.h: *50
cb_ConfigDev     pointer to struct ConfigDev in struct CurrentBinding
         +0x0000 libraries/configvars.h: *62
cb_FileName      pointer to unsigned char in struct CurrentBinding
         +0x0004 libraries/configvars.h: *63
cb_ProductString pointer to unsigned char in struct CurrentBinding
         +0x0008 libraries/configvars.h: *64
cb_ToolTypes     pointer to pointer to unsigned char in struct CurrentBinding
         +0x000c libraries/configvars.h: *65
cbh_CBport       struct MsgPort(size 0x0022 bytes) in struct ClipboardHandle
         +0x0034 libraries/iffparse.h: *118
cbh_Req          struct IOClipReq(size 0x0034 bytes) in struct ClipboardHandle
         +0x0000 libraries/iffparse.h: *117
cbh_SatisfyPort struct MsgPort(size 0x0022 bytes) in struct ClipboardHandle
         +0x0056 libraries/iffparse.h: *119
ccode            pointer to function returning int in struct Isrvstr
         +0x0016 graphics/graphint.h: *25
cd_BoardAddr     pointer to void in struct ConfigDev
         +0x0020 libraries/configvars.h: *39
cd_BoardSize     unsigned long int in struct ConfigDev
         +0x0024 libraries/configvars.h: *40
cd_Driver        pointer to void in struct ConfigDev
         +0x002c libraries/configvars.h: *43
cd_Flags         unsigned char in struct ConfigDev
         +0x000e libraries/configvars.h: *36
cd_NextCD        pointer to struct ConfigDev in struct ConfigDev
         +0x0030 libraries/configvars.h: *44
cd_Node          struct Node(size 0x000e bytes) in struct ConfigDev
         +0x0000 libraries/configvars.h: *35
cd_Pad           unsigned char in struct ConfigDev
         +0x000f libraries/configvars.h: *37
cd_Rom           struct ExpansionRom(size 0x0010 bytes) in struct ConfigDev
         +0x0010 libraries/configvars.h: *38
cd_SlotAddr      unsigned short int in struct ConfigDev
         +0x0028 libraries/configvars.h: *41
cd_SlotSize      unsigned short int in struct ConfigDev
         +0x002a libraries/configvars.h: *42
cd_Unused        array [4] of unsigned long int in struct ConfigDev
         +0x0034 libraries/configvars.h: *45
ceil             #define IEEEDPCeil  libraries/mathffp.h: *37
                 libraries/mathieeedp.h: *37
centphon         pointer to char in struct narrator_rb
         +0x0050 devices/narrator.h: *117
centralize       unsigned char in struct narrator_rb
         +0x004f devices/narrator.h: *116
cfc_ColorTable   pointer to unsigned short int in struct ColorFontColors
         +0x0004 graphics/text.h: *148
cfc_Count        unsigned short int in struct ColorFontColors
         +0x0002 graphics/text.h: *147
cfc_Reserved     unsigned short int in struct ColorFontColors
         +0x0000 graphics/text.h: *146
ch_masks         pointer to unsigned char in struct narrator_rb
         +0x0038 devices/narrator.h: *99
chanmask         unsigned char in struct narrator_rb
         +0x0043 devices/narrator.h: *104
check_lp         pointer to struct Layer in struct Layer_Info
         +0x0004 graphics/layers.h: *37
chm_ChangeCmd    long int in struct ClipHookMsg
         +0x0004 devices/clipboard.h: *67
chm_ClipID       long int in struct ClipHookMsg
         +0x0008 devices/clipboard.h: *69
```

Include File Cross Reference Page 18

```
chm_Type         unsigned long int in struct ClipHookMsg
         +0x0000 devices/clipboard.h: *66
ci_Data          pointer to unsigned char in struct CollectionItem
         +0x0008 libraries/iffparse.h: *108
ci_Next          pointer to struct CollectionItem in struct CollectionItem
         +0x0000 libraries/iffparse.h: *106
ci_Size          long int in struct CollectionItem
         +0x0004 libraries/iffparse.h: *107
cia              pointer to long int in struct GfxBase
         +0x002a graphics/gfxbase.h: *30
ciacra           unsigned char in struct CIA +0x0e00 hardware/cia.h: *60
ciacrb           unsigned char in struct CIA +0x0f00 hardware/cia.h: *62
ciaddra          unsigned char in struct CIA +0x0200 hardware/cia.h: *36
ciaddrb          unsigned char in struct CIA +0x0300 hardware/cia.h: *38
ciaicr           unsigned char in struct CIA +0x0d00 hardware/cia.h: *58
ciapra           unsigned char in struct CIA +0x0000 hardware/cia.h: *32
ciaprb           unsigned char in struct CIA +0x0100 hardware/cia.h: *34
ciasdr           unsigned char in struct CIA +0x0c00 hardware/cia.h: *56
ciatahi          unsigned char in struct CIA +0x0500 hardware/cia.h: *42
ciatalo          unsigned char in struct CIA +0x0400 hardware/cia.h: *40
ciatbhi          unsigned char in struct CIA +0x0700 hardware/cia.h: *46
ciatblo          unsigned char in struct CIA +0x0600 hardware/cia.h: *44
ciatodhi         unsigned char in struct CIA +0x0a00 hardware/cia.h: *52
ciatodlow        unsigned char in struct CIA +0x0800 hardware/cia.h: *48
ciatodmid        unsigned char in struct CIA +0x0900 hardware/cia.h: *50
cl_Dispatcher    struct Hook(size 0x0014 bytes) in struct IClass
         +0x0000 intuition/classes.h: *29
cl_Flags         unsigned long int in struct IClass
         +0x0030 intuition/classes.h: *43
cl_ID            pointer to unsigned char in struct IClass
         +0x001c intuition/classes.h: *32
cl_InstOffset    unsigned short int in struct IClass
         +0x0020 intuition/classes.h: *35
cl_InstSize      unsigned short int in struct IClass
         +0x0022 intuition/classes.h: *36
cl_ObjectCount   unsigned long int in struct IClass
         +0x002c intuition/classes.h: *41
cl_Reserved      unsigned long int in struct IClass
         +0x0014 intuition/classes.h: *30
cl_SubclassCount unsigned long int in struct IClass
         +0x0028 intuition/classes.h: *39
cl_Super         pointer to struct IClass in struct IClass
         +0x0018 intuition/classes.h: *31
cl_UserData      unsigned long int in struct IClass
         +0x0024 intuition/classes.h: *38
cleanup          pointer to function returning int in struct bltnode
         +0x000e hardware/blit.h: *97
cli_Background   long int in struct CommandLineInterface
         +0x002c dos/dosextens.h: *315
cli_CommandDir   long int in struct CommandLineInterface
         +0x0008 dos/dosextens.h: *306
cli_CommandFile long int in struct CommandLineInterface
         +0x0024 dos/dosextens.h: *313
cli_CommandName long int in struct CommandLineInterface
         +0x0010 dos/dosextens.h: *308
cli_CurrentInput long int in struct CommandLineInterface
         +0x0020 dos/dosextens.h: *312
cli_CurrentOutput long int in struct CommandLineInterface
         +0x0030 dos/dosextens.h: *316
cli_DefaultStack long int in struct CommandLineInterface
         +0x0034 dos/dosextens.h: *317
cli_FailLevel    long int in struct CommandLineInterface
         +0x0014 dos/dosextens.h: *309
cli_Interactive long int in struct CommandLineInterface
         +0x0028 dos/dosextens.h: *314
cli_Module       long int in struct CommandLineInterface
```

```
               +0x003c dos/dosextens.h: *319
cli_Prompt             long int in struct CommandLineInterface
               +0x0018 dos/dosextens.h: *310
cli_Result2            long int in struct CommandLineInterface
               +0x0000 dos/dosextens.h: *304
cli_ReturnCode         long int in struct CommandLineInterface
               +0x000c dos/dosextens.h: *307
cli_SetName            long int in struct CommandLineInterface
               +0x0004 dos/dosextens.h: *305
cli_StandardInput long int in struct CommandLineInterface
               +0x001c dos/dosextens.h: *311
cli_StandardOutput long int in struct CommandLineInterface
               +0x0038 dos/dosextens.h: *318
clxcon                 unsigned short int in struct Custom
               +0x0098 hardware/custom.h: *93
clxdat                 unsigned short int in struct Custom
               +0x000e hardware/custom.h: *35
cm_batch_items         pointer to struct TagItem in struct ColorMap
               +0x0020 graphics/view.h: *128
cm_vp                  pointer to struct ViewPort in struct ColorMap
               +0x0014 graphics/view.h: *125
cm_vpe                 pointer to struct ViewPortExtra in struct ColorMap
               +0x0008 graphics/view.h: *120
cn_ID                  long int in struct ContextNode
               +0x0008 libraries/iffparse.h: *71
cn_Node                struct MinNode(size 0x0008 bytes) in struct ContextNode
               +0x0000 libraries/iffparse.h: *70
cn_Scan                long int in struct ContextNode
               +0x0014 libraries/iffparse.h: *74
cn_Size                long int in struct ContextNode
               +0x0010 libraries/iffparse.h: *73
cn_Type                long int in struct ContextNode
               +0x000c libraries/iffparse.h: *72
code                   pointer to function returning int in struct Isrvstr
               +0x0012 graphics/graphint.h: *24
collHandler            pointer to struct collTable in struct GelsInfo
               +0x0012 graphics/rastport.h: *51
collPtrs               array [16] of pointer to function returning int in struct
                       collTable
               +0x0000 graphics/gels.h: *267
collTable              structure tag size 0x0040 graphics/rastport.h: *51
                       graphics/gels.h: 265
color                  array [32] of unsigned short int in struct Custom
               +0x0180 hardware/custom.h: *122
color0                 unsigned short int in struct Preferences
               +0x006e intuition/preferences.h: *70
color1                 unsigned short int in struct Preferences
               +0x0070 intuition/preferences.h: *71
color17                unsigned short int in struct Preferences
               +0x0066 intuition/preferences.h: *64
color18                unsigned short int in struct Preferences
               +0x0068 intuition/preferences.h: *65
color19                unsigned short int in struct Preferences
               +0x006a intuition/preferences.h: *66
color2                 unsigned short int in struct Preferences
               +0x0072 intuition/preferences.h: *72
color3                 unsigned short int in struct Preferences
               +0x0074 intuition/preferences.h: *73
colorByte              array [4] of unsigned char in union colorEntry
               +0x0000 devices/prtgfx.h: *30
colorEntry             union tag
          size 0x0004 devices/prtgfx.h: *28, 40, 41, 42, 43, 44
colorLong              unsigned long int in union colorEntry
               +0x0000 devices/prtgfx.h: *29
colorSByte             array [4] of char in union colorEntry
               +0x0000 devices/prtgfx.h: *31
```

```
control_delta_ntsc short int in struct GfxBase +0x017a graphics/gfxbase.h: *84
control_delta_pal short int in struct GfxBase +0x0178 graphics/gfxbase.h: *83
cop1lc                 unsigned long int in struct Custom
               +0x0080 hardware/custom.h: *83
cop2lc                 unsigned long int in struct Custom
               +0x0084 hardware/custom.h: *84
copcon                 unsigned short int in struct Custom
               +0x002e hardware/custom.h: *50
copinit                structure tag size 0x0078 graphics/copper.h: *92
                       graphics/gfxbase.h: 29
copinit                pointer to struct copinit in struct GfxBase
               +0x0026 graphics/gfxbase.h: *29
copins                 unsigned short int in struct Custom
               +0x008c hardware/custom.h: *87
copjmp1                unsigned short int in struct Custom
               +0x0088 hardware/custom.h: *85
copjmp2                unsigned short int in struct Custom
               +0x008a hardware/custom.h: *86
cos                    #define IEEEDPCos  libraries/mathffp.h: *41
                       libraries/mathieeedp.h: *41
cosh                   #define IEEEDPCosh  libraries/mathffp.h: *52
                       libraries/mathieeedp.h: *52
cp_x                   short int in struct RastPort +0x0024 graphics/rastport.h: *74
cp_y                   short int in struct RastPort +0x0026 graphics/rastport.h: *74
cpl_Array              pointer to pointer to struct MsgPort in struct CliProcList
               +0x000c dos/dosextens.h: *267
cpl_First              long int in struct CliProcList +0x0008 dos/dosextens.h: *266
cpl_Node               struct MinNode(size 0x0008 bytes) in struct CliProcList
               +0x0000 dos/dosextens.h: *265
cprlist                structure tag size 0x000a graphics/copper.h: *56, 58
                       graphics/view.h: 61, 62
cr                     pointer to struct ClipRect in struct Layer
               +0x0030 graphics/clip.h: *48
cr2                    pointer to struct ClipRect in struct Layer
               +0x0034 graphics/clip.h: *48
crb_reserved           array [5] of unsigned char in struct GfxBase
               +0x00ed graphics/gfxbase.h: *68
crnew                  pointer to struct ClipRect in struct Layer
               +0x0038 graphics/clip.h: *48
ctf_CharData           array [8] of pointer to void in struct ColorTextFont
               +0x0040 graphics/text.h: *162
ctf_ColorFontColors pointer to struct ColorFontColors in struct ColorTextFont
               +0x003c graphics/text.h: *161
ctf_Depth              unsigned char in struct ColorTextFont
               +0x0036 graphics/text.h: *155
ctf_FgColor            unsigned char in struct ColorTextFont
               +0x0037 graphics/text.h: *156
ctf_Flags              unsigned short int in struct ColorTextFont
               +0x0034 graphics/text.h: *154
ctf_High               unsigned char in struct ColorTextFont
               +0x0039 graphics/text.h: *158
ctf_Low                unsigned char in struct ColorTextFont
               +0x0038 graphics/text.h: *157
ctf_PlaneOnOff         unsigned char in struct ColorTextFont
               +0x003b graphics/text.h: *160
ctf_PlanePick          unsigned char in struct ColorTextFont
               +0x003a graphics/text.h: *159
ctf_TF                 struct TextFont(size 0x0034 bytes) in struct ColorTextFont
               +0x0000 graphics/text.h: *153
ctl                    unsigned short int in struct SpriteDef
               +0x0002 hardware/custom.h: *118
cu_AOLPen              char in struct ConUnit +0x0105 devices/conunit.h: *84
cu_AlgoStyle           unsigned char in struct ConUnit +0x0118 devices/conunit.h: *90
cu_BgPen               char in struct ConUnit +0x0104 devices/conunit.h: *83
cu_DrawMode            char in struct ConUnit +0x0106 devices/conunit.h: *85
cu_FgPen               char in struct ConUnit +0x0103 devices/conunit.h: *82
```

Include File Cross Reference Page 21

```
cu_Font          pointer to struct TextFont in struct ConUnit
         +0x0114 devices/conunit.h: *89
cu_KeyMapStruct struct KeyMap(size 0x0020 bytes) in struct ConUnit
         +0x0042 devices/conunit.h: *76
cu_MP            struct MsgPort(size 0x0022 bytes) in struct ConUnit
         +0x0000 devices/conunit.h: *56
cu_Mask          char in struct ConUnit +0x0102 devices/conunit.h: *81
cu_Minterms      array [8] of unsigned char in struct ConUnit
         +0x010c devices/conunit.h: *88
cu_Modes         array [3] of unsigned char in struct ConUnit
         +0x0122 devices/conunit.h: *98
cu_Node          struct Node(size 0x000e bytes) in struct ClipboardUnitPartial
         +0x0000 devices/clipboard.h: *37
cu_Obsolete1     char in struct ConUnit +0x0107 devices/conunit.h: *86
cu_Obsolete2     pointer to void in struct ConUnit
         +0x0108 devices/conunit.h: *87
cu_RawEvents     array [3] of unsigned char in struct ConUnit
         +0x0125 devices/conunit.h: *99
cu_TabStops      array [80] of unsigned short int in struct ConUnit
         +0x0062 devices/conunit.h: *78
cu_TxBaseline    unsigned short int in struct ConUnit
         +0x011e devices/conunit.h: *94
cu_TxFlags       unsigned char in struct ConUnit +0x0119 devices/conunit.h: *91
cu_TxHeight      unsigned short int in struct ConUnit
         +0x011a devices/conunit.h: *92
cu_TxSpacing     short int in struct ConUnit +0x0120 devices/conunit.h: *95
cu_TxWidth       unsigned short int in struct ConUnit
         +0x011c devices/conunit.h: *93
cu_UnitNum       unsigned long int in struct ClipboardUnitPartial
         +0x000e devices/clipboard.h: *38
cu_Window        pointer to struct Window in struct ConUnit
         +0x0022 devices/conunit.h: *58
cu_XCCP          short int in struct ConUnit +0x003e devices/conunit.h: *71
cu_XCP           short int in struct ConUnit +0x0026 devices/conunit.h: *59
cu_XMax          short int in struct ConUnit +0x002a devices/conunit.h: *61
cu_XMinShrink    short int in struct ConUnit +0x003a devices/conunit.h: *69
cu_XRExtant      short int in struct ConUnit +0x0036 devices/conunit.h: *67
cu_XROrigin      short int in struct ConUnit +0x0032 devices/conunit.h: *65
cu_XRSize        short int in struct ConUnit +0x002e devices/conunit.h: *63
cu_YCCP          short int in struct ConUnit +0x0040 devices/conunit.h: *72
cu_YCP           short int in struct ConUnit +0x0028 devices/conunit.h: *60
cu_YMax          short int in struct ConUnit +0x002c devices/conunit.h: *62
cu_YMinShrink    short int in struct ConUnit +0x003c devices/conunit.h: *70
cu_YRExtant      short int in struct ConUnit +0x0038 devices/conunit.h: *68
cu_YROrigin      short int in struct ConUnit +0x0034 devices/conunit.h: *66
cu_YRSize        short int in struct ConUnit +0x0030 devices/conunit.h: *64
current_monitor pointer to struct MonitorSpec in struct GfxBase
         +0x017c graphics/gfxbase.h: *85
current_tot_cclks unsigned short int in struct GfxBase
         +0x0160 graphics/gfxbase.h: *78
current_tot_rows unsigned short int in struct GfxBase
         +0x015e graphics/gfxbase.h: *77
CACRF_ClearD     #define (1L<<11) = 0x00000800  exec/execbase.h: *179
CACRF_ClearI     #define (1L<<3) = 0x00000008  exec/execbase.h: *175
CACRF_CopyBack   #define (1L<<31) = 0x80000000  exec/execbase.h: *183
CACRF_DBE        #define (1L<<12) = 0x00001000  exec/execbase.h: *180
CACRF_EnableD    #define (1L<<8) = 0x00000100  exec/execbase.h: *177
CACRF_EnableI    #define (1L<<0) = 0x00000001  exec/execbase.h: *173
CACRF_FreezeD    #define (1L<<9) = 0x00000200  exec/execbase.h: *178
CACRF_FreezeI    #define (1L<<1) = 0x00000002  exec/execbase.h: *174
CACRF_IBE        #define (1L<<4) = 0x00000010  exec/execbase.h: *176
CACRF_WriteAllocate #define (1L<<13) = 0x00002000  exec/execbase.h: *182
CBD_CHANGEHOOK   #define (CMD_NONSTD+3) = 0x0000000c  devices/clipboard.h: *31
CBD_CURRENTREADID #define (CMD_NONSTD+1) = 0x0000000a
                 devices/clipboard.h: *29
CBD_CURRENTWRITEID #define (CMD_NONSTD+2) = 0x0000000b
```

Include File Cross Reference Page 22

```
                 devices/clipboard.h: *30
CBD_DESCRLEN     #define 40 = 0x00000028  libraries/commodities.h: *38
CBD_NAMELEN      #define 24 = 0x00000018  libraries/commodities.h: *36
CBD_POST         #define (CMD_NONSTD+0) = 0x00000009  devices/clipboard.h: *28
CBD_TITLELEN     #define 40 = 0x00000028  libraries/commodities.h: *37
CBERR_DUP        #define 2 = 0x00000002  libraries/commodities.h: *43
CBERR_OBSOLETEID #define 1 = 0x00000001  devices/clipboard.h: *33
CBERR_OK         #define 0 = 0x00000000  libraries/commodities.h: *41
CBERR_SYSERR     #define 1 = 0x00000001  libraries/commodities.h: *42
CBERR_VERSION    #define 3 = 0x00000003  libraries/commodities.h: *44
CBM_MPS1000      #define 0x03 = 0x00000003  intuition/preferences.h: *195
CDB_BADMEMORY    #define 2 = 0x00000002  libraries/configvars.h: *51
CDB_CONFIGME     #define 1 = 0x00000001  libraries/configvars.h: *50
CDB_SHUTUP       #define 0 = 0x00000000  libraries/configvars.h: *49
CDF_BADMEMORY    #define 0x04 = 0x00000004  libraries/configvars.h: *55
CDF_CONFIGME     #define 0x02 = 0x00000002  libraries/configvars.h: *54
CDF_SHUTUP       #define 0x01 = 0x00000001  libraries/configvars.h: *53
CD_ASKDEFAULTKEYMAP #define (CMD_NONSTD+2) = 0x0000000b
                 devices/console.h: *26
CD_ASKKEYMAP     #define (CMD_NONSTD+0) = 0x00000009  devices/console.h: *24
CD_SETDEFAULTKEYMAP #define (CMD_NONSTD+3) = 0x0000000c
                 devices/console.h: *27
CD_SETKEYMAP     #define (CMD_NONSTD+1) = 0x0000000a  devices/console.h: *25
CEND             macro (1 argument)   graphics/gfxmacros.h: *41
CENTER_IMAGE     #define 0x0008 = 0x00000008  intuition/preferences.h: *245
CHANGEWINDOW     #define IDCMP_CHANGEWINDOW = 0x02000000
                 intuition/iobsolete.h: *139
CHANGE_FH        #define 1 = 0x00000001  dos/dos.h: *223
CHANGE_LOCK      #define 0 = 0x00000000  dos/dos.h: *222
CHECKBOXIDCMP    #define (IDCMP_GADGETUP) = 0x00000040
                 libraries/gadtools.h: *67
CHECKBOX_KIND    #define 2 = 0x00000002  libraries/gadtools.h: *36
CHECKED          #define 0x0100 = 0x00000100  intuition/intuition.h: *132
CHECKIMAGE       #define (0x0EL) = 0x0000000e  intuition/imageclass.h: *110
CHECKIT          #define 0x0001 = 0x00000001  intuition/intuition.h: *118
CHECKWIDTH       #define 19 = 0x00000013  intuition/intuition.h: *1300
CHeight          unsigned short int in struct PropInfo
         +0x000c intuition/intuition.h: *489
CIA              structure tag size 0x0f01 hardware/cia.h: *31
CIAANAME         #define "ciaa.resource"  resources/cia.h: *15
CIABNAME         #define "ciab.resource"  resources/cia.h: *16
CIAB_COMCD       #define (5) = 0x00000005  hardware/cia.h: *149
CIAB_COMCTS      #define (4) = 0x00000004  hardware/cia.h: *150
CIAB_COMDSR      #define (3) = 0x00000003  hardware/cia.h: *151
CIAB_COMDTR      #define (7) = 0x00000007  hardware/cia.h: *147
CIAB_COMRTS      #define (6) = 0x00000006  hardware/cia.h: *148
CIAB_DSKCHANGE   #define (2) = 0x00000002  hardware/cia.h: *140
CIAB_DSKDIREC    #define (1) = 0x00000001  hardware/cia.h: *163
CIAB_DSKMOTOR    #define (7) = 0x00000007  hardware/cia.h: *157
CIAB_DSKPROT     #define (3) = 0x00000003  hardware/cia.h: *139
CIAB_DSKRDY      #define (5) = 0x00000005  hardware/cia.h: *137
CIAB_DSKSEL0     #define (3) = 0x00000003  hardware/cia.h: *161
CIAB_DSKSEL1     #define (4) = 0x00000004  hardware/cia.h: *160
CIAB_DSKSEL2     #define (5) = 0x00000005  hardware/cia.h: *159
CIAB_DSKSEL3     #define (6) = 0x00000006  hardware/cia.h: *158
CIAB_DSKSIDE     #define (2) = 0x00000002  hardware/cia.h: *162
CIAB_DSKSTEP     #define (0) = 0x00000000  hardware/cia.h: *164
CIAB_DSKTRACK0   #define (4) = 0x00000004  hardware/cia.h: *138
CIAB_GAMEPORT0   #define (6) = 0x00000006  hardware/cia.h: *136
CIAB_GAMEPORT1   #define (7) = 0x00000007  hardware/cia.h: *135
CIAB_LED         #define (1) = 0x00000001  hardware/cia.h: *141
CIAB_OVERLAY     #define (0) = 0x00000000  hardware/cia.h: *142
CIAB_PRTRBUSY    #define (0) = 0x00000000  hardware/cia.h: *154
CIAB_PRTRPOUT    #define (1) = 0x00000001  hardware/cia.h: *153
CIAB_PRTRSEL     #define (2) = 0x00000002  hardware/cia.h: *152
CIACRAB_INMODE   #define 5 = 0x00000005  hardware/cia.h: *81
```

Include File Cross Reference Page 23

```
CIACRAB_LOAD       #define 4 = 0x00000004  hardware/cia.h: *80
CIACRAB_OUTMODE    #define 2 = 0x00000002  hardware/cia.h: *78
CIACRAB_PBON       #define 1 = 0x00000001  hardware/cia.h: *77
CIACRAB_RUNMODE    #define 3 = 0x00000003  hardware/cia.h: *79
CIACRAB_SPMODE     #define 6 = 0x00000006  hardware/cia.h: *82
CIACRAB_START      #define 0 = 0x00000000  hardware/cia.h: *76
CIACRAB_TODIN      #define 7 = 0x00000007  hardware/cia.h: *83
CIACRAF_INMODE     #define (1<<CIACRAB_INMODE) = 0x00000020
                   hardware/cia.h: *110
CIACRAF_LOAD       #define (1<<CIACRAB_LOAD) = 0x00000010  hardware/cia.h: *109
CIACRAF_OUTMODE    #define (1<<CIACRAB_OUTMODE) = 0x00000004
                   hardware/cia.h: *107
CIACRAF_PBON       #define (1<<CIACRAB_PBON) = 0x00000002  hardware/cia.h: *106
CIACRAF_RUNMODE    #define (1<<CIACRAB_RUNMODE) = 0x00000008
                   hardware/cia.h: *108
CIACRAF_SPMODE     #define (1<<CIACRAB_SPMODE) = 0x00000040
                   hardware/cia.h: *111
CIACRAF_START      #define (1<<CIACRAB_START) = 0x00000001  hardware/cia.h: *105
CIACRAF_TODIN      #define (1<<CIACRAB_TODIN) = 0x00000080  hardware/cia.h: *112
CIACRBB_ALARM      #define 7 = 0x00000007  hardware/cia.h: *93
CIACRBB_INMODE0    #define 5 = 0x00000005  hardware/cia.h: *91
CIACRBB_INMODE1    #define 6 = 0x00000006  hardware/cia.h: *92
CIACRBB_LOAD       #define 4 = 0x00000004  hardware/cia.h: *90
CIACRBB_OUTMODE    #define 2 = 0x00000002  hardware/cia.h: *88
CIACRBB_PBON       #define 1 = 0x00000001  hardware/cia.h: *87
CIACRBB_RUNMODE    #define 3 = 0x00000003  hardware/cia.h: *89
CIACRBB_START      #define 0 = 0x00000000  hardware/cia.h: *86
CIACRBF_ALARM      #define (1<<CIACRBB_ALARM) = 0x00000080  hardware/cia.h: *122
CIACRBF_INMODE0    #define (1<<CIACRBB_INMODE0) = 0x00000020
                   hardware/cia.h: *120
CIACRBF_INMODE1    #define (1<<CIACRBB_INMODE1) = 0x00000040
                   hardware/cia.h: *121
CIACRBF_IN_CNT     #define (CIACRBF_INMODE0) = 0x00000020  hardware/cia.h: *126
CIACRBF_IN_CNT_TA #define (CIACRBF_INMODE0|CIACRBF_INMODE1) = 0x00000060
                   hardware/cia.h: *128
CIACRBF_IN_PHI2    #define 0 = 0x00000000  hardware/cia.h: *125
CIACRBF_IN_TA      #define (CIACRBF_INMODE1) = 0x00000040  hardware/cia.h: *127
CIACRBF_LOAD       #define (1<<CIACRBB_LOAD) = 0x00000010  hardware/cia.h: *119
CIACRBF_OUTMODE    #define (1<<CIACRBB_OUTMODE) = 0x00000004
                   hardware/cia.h: *117
CIACRBF_PBON       #define (1<<CIACRBB_PBON) = 0x00000002  hardware/cia.h: *116
CIACRBF_RUNMODE    #define (1<<CIACRBB_RUNMODE) = 0x00000008
                   hardware/cia.h: *118
CIACRBF_START      #define (1<<CIACRBB_START) = 0x00000001  hardware/cia.h: *115
CIAF_COMCD         #define (1<<5) = 0x00000020  hardware/cia.h: *181
CIAF_COMCTS        #define (1<<4) = 0x00000010  hardware/cia.h: *182
CIAF_COMDSR        #define (1<<3) = 0x00000008  hardware/cia.h: *183
CIAF_COMDTR        #define (1<<7) = 0x00000080  hardware/cia.h: *179
CIAF_COMRTS        #define (1<<6) = 0x00000040  hardware/cia.h: *180
CIAF_DSKCHANGE     #define (1<<2) = 0x00000004  hardware/cia.h: *172
CIAF_DSKDIREC      #define (1<<1) = 0x00000002  hardware/cia.h: *195
CIAF_DSKMOTOR      #define (1<<7) = 0x00000080  hardware/cia.h: *189
CIAF_DSKPROT       #define (1<<3) = 0x00000008  hardware/cia.h: *171
CIAF_DSKRDY        #define (1<<5) = 0x00000020  hardware/cia.h: *169
CIAF_DSKSEL0       #define (1<<3) = 0x00000008  hardware/cia.h: *193
CIAF_DSKSEL1       #define (1<<4) = 0x00000010  hardware/cia.h: *192
CIAF_DSKSEL2       #define (1<<5) = 0x00000020  hardware/cia.h: *191
CIAF_DSKSEL3       #define (1<<6) = 0x00000040  hardware/cia.h: *190
CIAF_DSKSIDE       #define (1<<2) = 0x00000004  hardware/cia.h: *194
CIAF_DSKSTEP       #define (1<<0) = 0x00000001  hardware/cia.h: *196
CIAF_DSKTRACK0     #define (1<<4) = 0x00000010  hardware/cia.h: *170
CIAF_GAMEPORT0     #define (1<<6) = 0x00000040  hardware/cia.h: *168
CIAF_GAMEPORT1     #define (1<<7) = 0x00000080  hardware/cia.h: *167
CIAF_LED           #define (1<<1) = 0x00000002  hardware/cia.h: *173
CIAF_OVERLAY       #define (1<<0) = 0x00000001  hardware/cia.h: *174
CIAF_PRTRBUSY      #define (1<<0) = 0x00000001  hardware/cia.h: *186
```

Include File Cross Reference Page 24

```
CIAF_PRTRPOUT      #define (1<<1) = 0x00000002  hardware/cia.h: *185
CIAF_PRTRSEL       #define (1<<2) = 0x00000004  hardware/cia.h: *184
CIAICRB_ALRM       #define 2 = 0x00000002  hardware/cia.h: *69
CIAICRB_FLG        #define 4 = 0x00000004  hardware/cia.h: *71
CIAICRB_IR         #define 7 = 0x00000007  hardware/cia.h: *72
CIAICRB_SETCLR     #define 7 = 0x00000007  hardware/cia.h: *73
CIAICRB_SP         #define 3 = 0x00000003  hardware/cia.h: *70
CIAICRB_TA         #define 0 = 0x00000000  hardware/cia.h: *67
CIAICRB_TB         #define 1 = 0x00000001  hardware/cia.h: *68
CIAICRF_ALRM       #define (1<<CIAICRB_ALRM) = 0x00000004  hardware/cia.h: *98
CIAICRF_FLG        #define (1<<CIAICRB_FLG) = 0x00000010  hardware/cia.h: *100
CIAICRF_IR         #define (1<<CIAICRB_IR) = 0x00000080  hardware/cia.h: *101
CIAICRF_SETCLR     #define (1<<CIAICRB_SETCLR) = 0x00000080
                   hardware/cia.h: *102
CIAICRF_SP         #define (1<<CIAICRB_SP) = 0x00000008  hardware/cia.h: *99
CIAICRF_TA         #define (1<<CIAICRB_TA) = 0x00000001  hardware/cia.h: *96
CIAICRF_TB         #define (1<<CIAICRB_TB) = 0x00000002  hardware/cia.h: *97
CINIT              macro (2 arguments)  graphics/gfxmacros.h: *38
CLEANME            #define CLEANUP = 0x00000040  hardware/blit.h: *102
CLEANUP            #define 0x40 = 0x00000040  hardware/blit.h: *101
CLF_INLIST         #define 0x00000001 = 0x00000001  intuition/classes.h: *44
CLIB_MACROS_H      #define  clib/macros.h: *2
CLOSE              #define GTYP_CLOSE = 0x00000080  intuition/iobsolete.h: *103
CLOSEGADGET        #define 3 = 0x00000003  intuition/intuitionbase.h: *50
CLOSEIMAGE         #define (0x03L) = 0x00000003  intuition/imageclass.h: *104
CLOSEWINDOW        #define IDCMP_CLOSEWINDOW = 0x00000200
                   intuition/iobsolete.h: *123
CLVALUE            macro (1 argument)  rexx/rexxio.h: *58
CLeft              short int in struct StringInfo
          +0x0014 intuition/intuition.h: *537
CMDE_NOBROKER      #define (-1) = 0xffffffff  libraries/commodities.h: *137
CMDE_NOMEM         #define (-3) = 0xfffffffd  libraries/commodities.h: *139
CMDE_NOPORT        #define (-2) = 0xfffffffe  libraries/commodities.h: *138
CMDE_OK            #define (0) = 0x00000000  libraries/commodities.h: *136
CMD_CLEAR          #define 5 = 0x00000005  exec/io.h: *56
CMD_DISABLED       #define -999 = 0xfffffc19  dos/dosextens.h: *297
CMD_FLUSH          #define 8 = 0x00000008  exec/io.h: *59
CMD_INTERNAL       #define -2 = 0xfffffffe  dos/dosextens.h: *296
CMD_INVALID        #define 0 = 0x00000000  exec/io.h: *51
CMD_NONSTD         #define 9 = 0x00000009  exec/io.h: *61
CMD_READ           #define 2 = 0x00000002  exec/io.h: *53
CMD_RESET          #define 1 = 0x00000001  exec/io.h: *52
CMD_START          #define 7 = 0x00000007  exec/io.h: *58
CMD_STOP           #define 6 = 0x00000006  exec/io.h: *57
CMD_SYSTEM         #define -1 = 0xffffffff  dos/dosextens.h: *295
CMD_UPDATE         #define 4 = 0x00000004  exec/io.h: *55
CMD_WRITE          #define 3 = 0x00000003  exec/io.h: *54
CMOVE              macro (3 arguments)  graphics/gfxmacros.h: *39
COERR_BADFILTER    #define 4 = 0x00000004  libraries/commodities.h: *163
COERR_BADTYPE      #define 8 = 0x00000008  libraries/commodities.h: *164
COERR_ISNULL       #define 1 = 0x00000001  libraries/commodities.h: *161
COERR_NULLATTACH #define 2 = 0x00000002  libraries/commodities.h: *162
COF_SHOW_HIDE      #define 4 = 0x00000004  libraries/commodities.h: *67
COLORMAP_TRANSPARENCY #define 0x01 = 0x00000001  graphics/view.h: *140
COLORMAP_TYPE_V1_2 #define 0x00 = 0x00000000  graphics/view.h: *135
COLORMAP_TYPE_V1_4 #define 0x01 = 0x00000001  graphics/view.h: *136
COLORMAP_TYPE_V36 #define COLORMAP_TYPE_V1_4 = 0x00000001
                   graphics/view.h: *137
COLORON            #define 0x0200 = 0x00000200  graphics/display.h: *21
COLORPLANE_TRANSPARENCY #define 0x02 = 0x00000002  graphics/view.h: *141
COMMSEQ            #define 0x0004 = 0x00000004  intuition/intuition.h: *120
COMMWIDTH          #define 27 = 0x0000001b  intuition/intuition.h: *1301
COMPLEMENT         #define 2 = 0x00000002  graphics/rastport.h: *96
COMPLEX_BIT        #define 1 = 0x00000001  dos/dosasl.h: *139
CONFLAG_DEFAULT    #define 0 = 0x00000000  devices/conunit.h: *46
CONFLAG_NODRAW_ON_NEWSIZE #define 1 = 0x00000001  devices/conunit.h: *47
```

Include File Cross Reference Page 25

```
CONU_CHARMAP     #define 1 = 0x00000001  devices/conunit.h: *41
CONU_LIBRARY     #define -1 = 0xffffffff  devices/conunit.h: *36
CONU_SNIPMAP     #define 3 = 0x00000003  devices/conunit.h: *42
CONU_STANDARD    #define 0 = 0x00000000  devices/conunit.h: *37
COPPER_MOVE      #define 0 = 0x00000000  graphics/copper.h: *19
COPPER_WAIT      #define 1 = 0x00000001  graphics/copper.h: *20
CORRECT_BLUE     #define 0x0004 = 0x00000004  intuition/preferences.h: *243
CORRECT_GREEN    #define 0x0002 = 0x00000002  intuition/preferences.h: *242
CORRECT_RED      #define 0x0001 = 0x00000001  intuition/preferences.h: *241
CORRECT_RGB_MASK #define (CORRECT_RED|CORRECT_GREEN|CORRECT_BLUE) = 0x00000007
                 intuition/preferences.h: *264
COUNT            short int  exec/types.h: *58
CPRNXTBUF        #define 2 = 0x00000002  graphics/copper.h: *21
CPR_NT_LOF       #define 0x8000 = 0x00008000  graphics/copper.h: *22
CPR_NT_SHT       #define 0x4000 = 0x00004000  graphics/copper.h: *23
CPR_NT_SYS       #define 0x2000 = 0x00002000  graphics/copper.h: *24
CPTR             typedef ULONG  exec/types.h: *60
CR_NEEDS_NO_CONCEALED_RASTERS #define 1 = 0x00000001  graphics/clip.h: *79
CR_NEEDS_NO_LAYERBLIT_DAMAGE #define 2 = 0x00000002  graphics/clip.h: *80
CSBLANK          #define 0x0008 = 0x00000008  hardware/custom.h: *156
CSCBLANKEN       #define 0x0400 = 0x00000400  hardware/custom.h: *149
CSYNCTRUE        #define 0x0004 = 0x00000004  hardware/custom.h: *157
CS_Buffer        pointer to unsigned char in struct CSource
         +0x0000 dos/rdargs.h: *64
CS_CurChr        long int in struct CSource +0x0008 dos/rdargs.h: *66
CS_Length        long int in struct CSource +0x0004 dos/rdargs.h: *65
CSource          structure tag size 0x000c dos/rdargs.h: *63, 98
CTB_ALPHA        #define 2 = 0x00000002  rexx/rxslib.h: *91
CTB_DIGIT        #define 1 = 0x00000001  rexx/rxslib.h: *90
CTB_LOWER        #define 7 = 0x00000007  rexx/rxslib.h: *96
CTB_MAPCOLOR     #define 0 = 0x00000000  graphics/text.h: *141
CTB_REXXOPR      #define 4 = 0x00000004  rexx/rxslib.h: *93
CTB_REXXSPC      #define 5 = 0x00000005  rexx/rxslib.h: *94
CTB_REXXSYM      #define 3 = 0x00000003  rexx/rxslib.h: *92
CTB_SPACE        #define 0 = 0x00000000  rexx/rxslib.h: *89
CTB_UPPER        #define 6 = 0x00000006  rexx/rxslib.h: *95
CTC_HCLRTAB      #define 2 = 0x00000002  devices/console.h: *90
CTC_HCLRTABSALL  #define 5 = 0x00000005  devices/console.h: *91
CTC_HSETTAB      #define 0 = 0x00000000  devices/console.h: *89
CTF_ALPHA        #define (1 << CTB_ALPHA) = 0x00000004  rexx/rxslib.h: *101
CTF_DIGIT        #define (1 << CTB_DIGIT) = 0x00000002  rexx/rxslib.h: *100
CTF_LOWER        #define (1 << CTB_LOWER) = 0x00000080  rexx/rxslib.h: *106
CTF_MAPCOLOR     #define 0x0001 = 0x00000001  graphics/text.h: *142
CTF_REXXOPR      #define (1 << CTB_REXXOPR) = 0x00000010  rexx/rxslib.h: *103
CTF_REXXSPC      #define (1 << CTB_REXXSPC) = 0x00000020  rexx/rxslib.h: *104
CTF_REXXSYM      #define (1 << CTB_REXXSYM) = 0x00000008  rexx/rxslib.h: *102
CTF_SPACE        #define (1 << CTB_SPACE) = 0x00000001  rexx/rxslib.h: *99
CTF_UPPER        #define (1 << CTB_UPPER) = 0x00000040  rexx/rxslib.h: *105
CT_ANTIALIAS     #define 0x0004 = 0x00000004  graphics/text.h: *139
CT_COLORFONT     #define 0x0001 = 0x00000001  graphics/text.h: *136
CT_COLORMASK     #define 0x000F = 0x0000000f  graphics/text.h: *135
CT_GREYFONT      #define 0x0002 = 0x00000002  graphics/text.h: *137
CTop             short int in struct StringInfo
         +0x0016 intuition/intuition.h: *537
CURSORDOWN       #define 0x4D = 0x0000004d  intuition/intuition.h: *1345
CURSORLEFT       #define 0x4F = 0x0000004f  intuition/intuition.h: *1343
CURSORRIGHT      #define 0x4E = 0x0000004e  intuition/intuition.h: *1344
CURSORUP         #define 0x4C = 0x0000004c  intuition/intuition.h: *1342
CUSTOM           #define 0x40 = 0x00000040  intuition/preferences.h: *189
CUSTOMBITMAP     #define 0x0040 = 0x00000040  intuition/screens.h: *169
CUSTOMGADGET     #define GTYP_CUSTOMGADGET = 0x00000005
                 intuition/iobsolete.h: *108
CUSTOMIMAGEDEPTH #define (-1) = 0xffffffff  intuition/imageclass.h: *24
CUSTOMSCREEN     #define 0x000F = 0x0000000f  intuition/screens.h: *163
CUSTOM_HOOK      macro (1 argument)   intuition/cghooks.h: *79
CUSTOM_NAME      #define 0x00 = 0x00000000  intuition/preferences.h: *192
```

Include File Cross Reference Page 26

```
CWAIT            macro (3 arguments)   graphics/gfxmacros.h: *40
CWidth           unsigned short int in struct PropInfo
         +0x000a intuition/intuition.h: *488
CXCMD_APPEAR     #define (19) = 0x00000013  libraries/commodities.h: *126
CXCMD_DISABLE    #define (15) = 0x0000000f  libraries/commodities.h: *124
CXCMD_DISAPPEAR  #define (21) = 0x00000015  libraries/commodities.h: *127
CXCMD_ENABLE     #define (17) = 0x00000011  libraries/commodities.h: *125
CXCMD_KILL       #define (23) = 0x00000017  libraries/commodities.h: *128
CXCMD_LIST_CHG   #define (27) = 0x0000001b  libraries/commodities.h: *132
CXCMD_UNIQUE     #define (25) = 0x00000019  libraries/commodities.h: *131
CXM_COMMAND      #define (1 << 6) = 0x00000040  libraries/commodities.h: *121
CXM_IEVENT       #define (1 << 5) = 0x00000020  libraries/commodities.h: *113
CXM_UNIQUE       #define (1 << 4) = 0x00000010  libraries/commodities.h: *99
CX_BROKER        #define 6 = 0x00000006  libraries/commodities.h: *91
CX_CUSTOM        #define 8 = 0x00000008  libraries/commodities.h: *93
CX_DEBUG         #define 7 = 0x00000007  libraries/commodities.h: *92
CX_FILTER        #define 1 = 0x00000001  libraries/commodities.h: *86
CX_INVALID       #define 0 = 0x00000000  libraries/commodities.h: *85
CX_SEND          #define 3 = 0x00000003  libraries/commodities.h: *88
CX_SIGNAL        #define 4 = 0x00000004  libraries/commodities.h: *89
CX_TRANSLATE     #define 5 = 0x00000005  libraries/commodities.h: *90
CX_TYPEFILTER    #define 2 = 0x00000002  libraries/commodities.h: *87
CX_ZERO          #define 9 = 0x00000009  libraries/commodities.h: *94
CYCLEIDCMP       #define (IDCMP_GADGETUP) = 0x00000040
                 libraries/gadtools.h: *74
CYCLE_KIND       #define 7 = 0x00000007  libraries/gadtools.h: *41
Carg             int in struct Isrvstr +0x001a graphics/graphint.h: *26
CheckMark        pointer to struct Image in struct Window
         +0x0064 intuition/intuition.h: *865
CheckMark        pointer to struct Image in struct NewWindow
         +0x0016 intuition/intuition.h: *996
CheckMark        pointer to struct Image in struct ExtNewWindow
         +0x0016 intuition/intuition.h: *1054
ChipRevBits0     unsigned char in struct GfxBase
         +0x00ec graphics/gfxbase.h: *67
ChkBase          unsigned long int in struct ExecBase
         +0x0026 exec/execbase.h: *43
ChkSum           unsigned short int in struct ExecBase
         +0x0052 exec/execbase.h: *55
Class            unsigned long int in struct IntuiMessage
         +0x0014 intuition/intuition.h: *684
Class            "UBYTE"  intuition/classes.h: *45
ClassID          pointer to "UBYTE"  intuition/classusr.h: *23
                 intuition/classes.h: 32
ClearPath        pointer to struct VSprite in struct VSprite
         +0x000c graphics/gels.h: *86
CliProcList      structure tag size 0x0010 dos/dosextens.h: *264
ClipHookMsg      structure tag size 0x000c devices/clipboard.h: *65
ClipRect         structure tag
     size 0x0024 graphics/clip.h: *37, 44, 48, 49, 50, 64, 66, 67,
                             71
                 graphics/layers.h: 38
ClipRect         pointer to struct ClipRect in struct Layer
         +0x0008 graphics/clip.h: *37
ClipRegion       pointer to struct Region in struct Layer
         +0x007e graphics/clip.h: *55
ClipboardHandle structure tag size 0x0078 libraries/iffparse.h: *116
ClipboardUnitPartial structure tag size 0x0012 devices/clipboard.h: *36, 46
Clock            long int in struct AnimOb +0x0008 graphics/gels.h: *211
ClockData        structure tag size 0x000e utility/date.h: *18
ClrIns           pointer to struct CopList in struct ViewPort
         +0x0010 graphics/view.h: *48
Code             unsigned short int in struct IntuiMessage
         +0x0018 intuition/intuition.h: *687
Code             unsigned short int in struct SGWork
         +0x0018 intuition/sghooks.h: *43
```

Include File Cross Reference Page 27

```
CoerceDisplayInfo pointer to void in struct ColorMap
        +0x001c graphics/view.h: *127
ColdCapture      pointer to void in struct ExecBase
        +0x002a exec/execbase.h: *44
CollMask         pointer to short int in struct VSprite
        +0x002c graphics/gels.h: *116
CollectionItem   structure tag size 0x000c libraries/iffparse.h: *105, 106
ColorFontColors structure tag size 0x0008 graphics/text.h: *145, 161
ColorIndex       short int in struct ColorSpec
        +0x0000 intuition/intuition.h: *1242
ColorMap         structure tag size 0x0028 devices/printer.h: *164
                 graphics/view.h: 44, 114
ColorMap         pointer to struct ColorMap in struct ViewPort
        +0x0004 graphics/view.h: *44
ColorSpec        structure tag size 0x0008 intuition/intuition.h: *1241
ColorTable       pointer to void in struct ColorMap
        +0x0004 graphics/view.h: *119
ColorTextFont    structure tag size 0x0060 graphics/text.h: *152
ColumnSizeChange char in struct Preferences
        +0x00d9 intuition/preferences.h: *115
Command          char in struct MenuItem +0x001a intuition/intuition.h: *106
CommandLineInterface structure tag size 0x0040 dos/dosextens.h: *303
Compatibility    short int in struct MonitorInfo
        +0x002a graphics/displayinfo.h: *119
ConUnit          structure tag size 0x0128 devices/conunit.h: *55
ConfigDev        structure tag size 0x0044 libraries/configvars.h: *34, 44, 62
ContextNode      structure tag size 0x0018 libraries/iffparse.h: *69
CoolCapture      pointer to void in struct ExecBase
        +0x002e exec/execbase.h: *45
CopIns           structure tag size 0x0006 graphics/copper.h: *26, 68, 69
CopIns           pointer to struct CopIns in struct CopList
        +0x000c graphics/copper.h: *68
CopLStart        pointer to unsigned short int in struct CopList
        +0x0014 graphics/copper.h: *70
CopList          structure tag
     size 0x0022 graphics/copper.h: *31, 63, 65, 66, 86, 87
                 graphics/view.h: 46, 47, 48
CopList          pointer to struct CopList in struct UCopList
        +0x0008 graphics/copper.h: *87
CopPtr           pointer to struct CopIns in struct CopList
        +0x0010 graphics/copper.h: *69
CopSStart        pointer to unsigned short int in struct CopList
        +0x0018 graphics/copper.h: *71
Count            short int in struct CopList +0x001c graphics/copper.h: *72
Count            unsigned short int in struct ColorMap
        +0x0002 graphics/view.h: *118
Count            short int in struct AreaInfo +0x0010 graphics/rastport.h: *29
Count            char in struct Border +0x0007 intuition/intuition.h: *603
CurrentBinding   structure tag size 0x0010 libraries/configvars.h: *61
                 libraries/expansionbase.h: 53
Custom           structure tag size 0x01e6 hardware/custom.h: *27
CustomBitMap     pointer to struct BitMap in struct NewScreen
        +0x001c intuition/screens.h: *332
CustomBitMap     pointer to struct BitMap in struct ExtNewScreen
        +0x001c intuition/screens.h: *355
CxCustom         macro (2 arguments)   libraries/commodities.h: *29
CxDebug          macro (1 argument)   libraries/commodities.h: *28
CxFilter         macro (1 argument)   libraries/commodities.h: *23
CxMsg            "LONG"  libraries/commodities.h: *76
CxObj            "LONG"  libraries/commodities.h: *75
CxSender         macro (2 arguments)   libraries/commodities.h: *25
CxSignal         macro (2 arguments)   libraries/commodities.h: *26
CxTranslate      macro (1 argument)   libraries/commodities.h: *27
CxTypeFilter     macro (1 argument)   libraries/commodities.h: *24
da_BootPoint     unsigned short int in struct DiagArea
        +0x0006 libraries/configregs.h: *242
```

Include File Cross Reference Page 28

```
da_Config        unsigned char in struct DiagArea
        +0x0000 libraries/configregs.h: *238
da_DiagPoint     unsigned short int in struct DiagArea
        +0x0004 libraries/configregs.h: *241
da_Flags         unsigned char in struct DiagArea
        +0x0001 libraries/configregs.h: *239
da_Name          unsigned short int in struct DiagArea
        +0x0008 libraries/configregs.h: *243
da_Reserved01    unsigned short int in struct DiagArea
        +0x000a libraries/configregs.h: *247
da_Reserved02    unsigned short int in struct DiagArea
        +0x000c libraries/configregs.h: *248
da_Size          unsigned short int in struct DiagArea
        +0x0002 libraries/configregs.h: *240
dat_Flags        unsigned char in struct DateTime +0x000d dos/datetime.h: *30
dat_Format       unsigned char in struct DateTime +0x000c dos/datetime.h: *29
dat_Stamp        struct DateStamp(size 0x000c bytes) in struct DateTime
        +0x0000 dos/datetime.h: *28
dat_StrDate      pointer to unsigned char in struct DateTime
        +0x0012 dos/datetime.h: *32
dat_StrDay       pointer to unsigned char in struct DateTime
        +0x000e dos/datetime.h: *31
dat_StrTime      pointer to unsigned char in struct DateTime
        +0x0016 dos/datetime.h: *33
dataa            unsigned short int in struct SpriteDef
        +0x0004 hardware/custom.h: *119
datab            unsigned short int in struct SpriteDef
        +0x0006 hardware/custom.h: *120
dbf              function returning "LONG"   libraries/mathffp.h: *78
dd_CmdBytes      pointer to void in struct DeviceData
        +0x002e devices/prtbase.h: *56
dd_CmdVectors    pointer to void in struct DeviceData
        +0x002a devices/prtbase.h: *55
dd_CurrentX      long int in struct OldDrawerData
        +0x0030 workbench/workbench.h: *46
dd_CurrentX      long int in struct DrawerData
        +0x0030 workbench/workbench.h: *54
dd_CurrentY      long int in struct OldDrawerData
        +0x0034 workbench/workbench.h: *47
dd_CurrentY      long int in struct DrawerData
        +0x0034 workbench/workbench.h: *55
dd_Device        struct Library(size 0x0022 bytes) in struct DeviceData
        +0x0000 devices/prtbase.h: *52
dd_ExecBase      pointer to void in struct DeviceData
        +0x0026 devices/prtbase.h: *54
dd_Flags         unsigned long int in struct DrawerData
        +0x0038 workbench/workbench.h: *56
dd_Library       struct Library(size 0x0022 bytes) in struct Device
        +0x0000 exec/devices.h: *27
dd_NewWindow     struct NewWindow(size 0x0030 bytes) in struct OldDrawerData
        +0x0000 workbench/workbench.h: *45
dd_NewWindow     struct NewWindow(size 0x0030 bytes) in struct DrawerData
        +0x0000 workbench/workbench.h: *53
dd_NumCommands   unsigned short int in struct DeviceData
        +0x0032 devices/prtbase.h: *57
dd_Segment       pointer to void in struct DeviceData
        +0x0022 devices/prtbase.h: *53
dd_ViewModes     unsigned short int in struct DrawerData
        +0x003c workbench/workbench.h: *57
ddfstop          unsigned short int in struct Custom
        +0x0094 hardware/custom.h: *91
ddfstrt          unsigned short int in struct Custom
        +0x0092 hardware/custom.h: *90
de_Baud          unsigned long int in struct DosEnvec
        +0x0044 dos/filehandler.h: *49
de_BlocksPerTrack unsigned long int in struct DosEnvec
```

Include File Cross Reference Page 29

```
          +0x0014 dos/filehandler.h: *35
de_BootBlocks     unsigned long int in struct DosEnvec
          +0x004c dos/filehandler.h: *51
de_BootPri        long int in struct DosEnvec +0x003c dos/filehandler.h: *45
de_BufMemType     unsigned long int in struct DosEnvec
          +0x0030 dos/filehandler.h: *42
de_Control        unsigned long int in struct DosEnvec
          +0x0048 dos/filehandler.h: *50
de_DosType        unsigned long int in struct DosEnvec
          +0x0040 dos/filehandler.h: *46
de_HighCyl        unsigned long int in struct DosEnvec
          +0x0028 dos/filehandler.h: *40
de_Interleave     unsigned long int in struct DosEnvec
          +0x0020 dos/filehandler.h: *38
de_LowCyl         unsigned long int in struct DosEnvec
          +0x0024 dos/filehandler.h: *39
de_Mask           unsigned long int in struct DosEnvec
          +0x0038 dos/filehandler.h: *44
de_MaxTransfer    unsigned long int in struct DosEnvec
          +0x0034 dos/filehandler.h: *43
de_NumBuffers     unsigned long int in struct DosEnvec
          +0x002c dos/filehandler.h: *41
de_PreAlloc       unsigned long int in struct DosEnvec
          +0x001c dos/filehandler.h: *37
de_Reserved       unsigned long int in struct DosEnvec
          +0x0018 dos/filehandler.h: *36
de_SecOrg         unsigned long int in struct DosEnvec
          +0x0008 dos/filehandler.h: *32
de_SectorPerBlock unsigned long int in struct DosEnvec
          +0x0010 dos/filehandler.h: *34
de_SizeBlock      unsigned long int in struct DosEnvec
          +0x0004 dos/filehandler.h: *31
de_Surfaces       unsigned long int in struct DosEnvec
          +0x000c dos/filehandler.h: *33
de_TableSize      unsigned long int in struct DosEnvec
          +0x0000 dos/filehandler.h: *30
default_monitor pointer to struct MonitorSpec in struct GfxBase
          +0x018e graphics/gfxbase.h: *87
deniseid          unsigned short int in struct Custom
          +0x007c hardware/custom.h: *81
detailPen         #define DETAILPEN = 0x00000000  intuition/iobsolete.h: *261
dfh_DF            struct Node(size 0x000e bytes) in struct DiskFontHeader
          +0x0000 libraries/diskfont.h: *72
dfh_FileID        unsigned short int in struct DiskFontHeader
          +0x000e libraries/diskfont.h: *73
dfh_Name          array [32] of char in struct DiskFontHeader
          +0x0016 libraries/diskfont.h: *76
dfh_Revision      unsigned short int in struct DiskFontHeader
          +0x0010 libraries/diskfont.h: *74
dfh_Segment       long int in struct DiskFontHeader
          +0x0012 libraries/diskfont.h: *75
dfh_TF            struct TextFont(size 0x0034 bytes) in struct DiskFontHeader
          +0x0036 libraries/diskfont.h: *77
dfh_TagList       #define dfh_Segment  libraries/diskfont.h: *82
dg_BufMemType     unsigned long int in struct DriveGeometry
          +0x0018 devices/trackdisk.h: *144
dg_CylSectors     unsigned long int in struct DriveGeometry
          +0x000c devices/trackdisk.h: *141
dg_Cylinders      unsigned long int in struct DriveGeometry
          +0x0008 devices/trackdisk.h: *140
dg_DeviceType     unsigned char in struct DriveGeometry
          +0x001c devices/trackdisk.h: *146
dg_Flags          unsigned char in struct DriveGeometry
          +0x001d devices/trackdisk.h: *147
dg_Heads          unsigned long int in struct DriveGeometry
          +0x0010 devices/trackdisk.h: *142
```

Include File Cross Reference Page 30

```
dg_Reserved       unsigned short int in struct DriveGeometry
          +0x001e devices/trackdisk.h: *148
dg_SectorSize     unsigned long int in struct DriveGeometry
          +0x0000 devices/trackdisk.h: *138
dg_TotalSectors unsigned long int in struct DriveGeometry
          +0x0004 devices/trackdisk.h: *139
dg_TrackSectors unsigned long int in struct DriveGeometry
          +0x0014 devices/trackdisk.h: *143
di_DevInfo        long int in struct DosInfo +0x0004 dos/dosextens.h: *277
di_DevLock        struct SignalSemaphore(size 0x002e bytes) in struct DosInfo
          +0x0014 dos/dosextens.h: *281
di_Devices        long int in struct DosInfo +0x0008 dos/dosextens.h: *278
di_EntryLock      struct SignalSemaphore(size 0x002e bytes) in struct DosInfo
          +0x0042 dos/dosextens.h: *282
di_Handlers       long int in struct DosInfo +0x000c dos/dosextens.h: *279
di_McName         long int in struct DosInfo +0x0000 dos/dosextens.h: *275
di_NetHand        pointer to void in struct DosInfo
          +0x0010 dos/dosextens.h: *280
di_ResList        #define di_McName  dos/dosextens.h: *276
diagstrt          array [4] of unsigned short int in struct copinit
          +0x000c graphics/copper.h: *96
diwhigh           unsigned short int in struct Custom
          +0x01e4 hardware/custom.h: *141
diwstart          array [4] of unsigned short int in struct copinit
          +0x0004 graphics/copper.h: *95
diwstop           unsigned short int in struct Custom
          +0x0090 hardware/custom.h: *89
diwstrt           unsigned short int in struct Custom
          +0x008e hardware/custom.h: *88
dl_A2             long int in struct DosLibrary +0x002a dos/dosextens.h: *232
dl_A5             long int in struct DosLibrary +0x002e dos/dosextens.h: *233
dl_A6             long int in struct DosLibrary +0x0032 dos/dosextens.h: *234
dl_DiskType       long int in struct DeviceList +0x0020 dos/dosextens.h: *338
dl_Errors         pointer to struct ErrorString in struct DosLibrary
          +0x0036 dos/dosextens.h: *235
dl_GV             pointer to void in struct DosLibrary
          +0x0026 dos/dosextens.h: *231
dl_Lock           long int in struct DeviceList +0x000c dos/dosextens.h: *335
dl_LockList       long int in struct DeviceList +0x001c dos/dosextens.h: *337
dl_Name           long int in struct DeviceList +0x0028 dos/dosextens.h: *340
dl_Next           long int in struct DeviceList +0x0000 dos/dosextens.h: *332
dl_Root           pointer to struct RootNode in struct DosLibrary
          +0x0022 dos/dosextens.h: *230
dl_Task           pointer to struct MsgPort in struct DeviceList
          +0x0008 dos/dosextens.h: *334
dl_TimeReq        pointer to struct timerequest in struct DosLibrary
          +0x003a dos/dosextens.h: *236
dl_Type           long int in struct DeviceList +0x0004 dos/dosextens.h: *333
dl_UtilityBase    pointer to struct Library in struct DosLibrary
          +0x003e dos/dosextens.h: *237
dl_VolumeDate     struct DateStamp(size 0x000c bytes) in struct DeviceList
          +0x0010 dos/dosextens.h: *336
dl_lib            struct Library(size 0x0022 bytes) in struct DosLibrary
          +0x0000 dos/dosextens.h: *229
dl_unused         long int in struct DeviceList +0x0024 dos/dosextens.h: *339
dmacon            unsigned short int in struct Custom
          +0x0096 hardware/custom.h: *92
dmaconr           unsigned short int in struct Custom
          +0x0002 hardware/custom.h: *29
dn_GlobalVec      long int in struct DeviceNode +0x0024 dos/filehandler.h: *114
dn_Handler        long int in struct DeviceNode +0x0010 dos/filehandler.h: *108
dn_Lock           long int in struct DeviceNode +0x000c dos/filehandler.h: *107
dn_Name           long int in struct DeviceNode +0x0028 dos/filehandler.h: *122
dn_Next           long int in struct DeviceNode +0x0000 dos/filehandler.h: *102
dn_Priority       long int in struct DeviceNode +0x0018 dos/filehandler.h: *110
dn_SegList        long int in struct DeviceNode +0x0020 dos/filehandler.h: *112
```

Include File Cross Reference Page 31

```
dn_StackSize     unsigned long int in struct DeviceNode
         +0x0014 dos/filehandler.h: *109
dn_Startup       long int in struct DeviceNode +0x001c dos/filehandler.h: *111
dn_Task          pointer to struct MsgPort in struct DeviceNode
         +0x0008 dos/filehandler.h: *104
dn_Type          unsigned long int in struct DeviceNode
         +0x0004 dos/filehandler.h: *103
do_CurrentX      long int in struct DiskObject
         +0x003a workbench/workbench.h: *69
do_CurrentY      long int in struct DiskObject
         +0x003e workbench/workbench.h: *70
do_DefaultTool   pointer to char in struct DiskObject
         +0x0032 workbench/workbench.h: *67
do_DrawerData    pointer to struct DrawerData in struct DiskObject
         +0x0042 workbench/workbench.h: *71
do_Gadget        struct Gadget(size 0x002c bytes) in struct DiskObject
         +0x0004 workbench/workbench.h: *65
do_Magic         unsigned short int in struct DiskObject
         +0x0000 workbench/workbench.h: *63
do_StackSize     long int in struct DiskObject
         +0x004a workbench/workbench.h: *73
do_ToolTypes     pointer to pointer to char in struct DiskObject
         +0x0036 workbench/workbench.h: *68
do_ToolWindow    pointer to char in struct DiskObject
         +0x0046 workbench/workbench.h: *72
do_Type          unsigned char in struct DiskObject
         +0x0030 workbench/workbench.h: *66
do_Version       unsigned short int in struct DiskObject
         +0x0002 workbench/workbench.h: *64
do_monitor       pointer to function returning int in struct SpecialMonitor
         +0x001a graphics/monitor.h: *146
dol_AssignName   pointer to unsigned char in struct (no tag)
         +0x0000 dos/dosextens.h: *385
dol_DiskType     long int in struct (no tag)  +0x0010 dos/dosextens.h: *381
dol_GlobVec      long int in struct (no tag)  +0x0014 dos/dosextens.h: *373
dol_Handler      long int in struct (no tag)  +0x0000 dos/dosextens.h: *368
dol_List         pointer to struct AssignList in struct (no tag)
         +0x0004 dos/dosextens.h: *386
dol_Lock         long int in struct DosList +0x000c dos/dosextens.h: *365
dol_LockList     long int in struct (no tag)  +0x000c dos/dosextens.h: *380
dol_Name         long int in struct DosList +0x0028 dos/dosextens.h: *391
dol_Next         long int in struct DosList +0x0000 dos/dosextens.h: *362
dol_Priority     long int in struct (no tag)  +0x0008 dos/dosextens.h: *370
dol_SegList      long int in struct (no tag)  +0x0010 dos/dosextens.h: *372
dol_StackSize    long int in struct (no tag)  +0x0004 dos/dosextens.h: *369
dol_Startup      unsigned long int in struct (no tag)
         +0x000c dos/dosextens.h: *371
dol_Task         pointer to struct MsgPort in struct DosList
         +0x0008 dos/dosextens.h: *364
dol_Type         long int in struct DosList +0x0004 dos/dosextens.h: *363
dol_VolumeDate   struct DateStamp(size 0x000c bytes) in struct (no tag)
         +0x0000 dos/dosextens.h: *379
dol_assign       struct (no tag) (size 0x0008 bytes) in union (no tag)
         +0x0000 dos/dosextens.h: *387
dol_handler      struct (no tag) (size 0x0018 bytes) in union (no tag)
         +0x0000 dos/dosextens.h: *376
dol_misc         union (no tag) (size 0x0018 bytes) in struct DosList
         +0x0010 dos/dosextens.h: *389
dol_volume       struct (no tag) (size 0x0014 bytes) in union (no tag)
         +0x0000 dos/dosextens.h: *382
dp_Action        #define dp_Type  dos/dosextens.h: *124
dp_Arg1          long int in struct DosPacket +0x0014 dos/dosextens.h: *128
dp_Arg2          long int in struct DosPacket +0x0018 dos/dosextens.h: *129
dp_Arg3          long int in struct DosPacket +0x001c dos/dosextens.h: *130
dp_Arg4          long int in struct DosPacket +0x0020 dos/dosextens.h: *131
dp_Arg5          long int in struct DosPacket +0x0024 dos/dosextens.h: *132
```

Include File Cross Reference Page 32

```
dp_Arg6          long int in struct DosPacket +0x0028 dos/dosextens.h: *133
dp_Arg7          long int in struct DosPacket +0x002c dos/dosextens.h: *134
dp_BufAddr       #define dp_Arg1  dos/dosextens.h: *127
dp_Link          pointer to struct Message in struct DosPacket
         +0x0000 dos/dosextens.h: *111
dp_Port          pointer to struct MsgPort in struct DosPacket
         +0x0004 dos/dosextens.h: *112
dp_Res1          long int in struct DosPacket +0x000c dos/dosextens.h: *117
dp_Res2          long int in struct DosPacket +0x0010 dos/dosextens.h: *121
dp_Status        #define dp_Res1  dos/dosextens.h: *125
dp_Status2       #define dp_Res2  dos/dosextens.h: *126
dp_Type          long int in struct DosPacket +0x0008 dos/dosextens.h: *114
dr_CiaResource   pointer to struct Library in struct DiscResource
         +0x002c resources/disk.h: *56
dr_CurrTask      pointer to struct Task in struct DiscResource
         +0x0090 resources/disk.h: *62
dr_Current       pointer to struct DiscResourceUnit in struct DiscResource
         +0x0022 resources/disk.h: *52
dr_DiscBlock     struct Interrupt(size 0x0016 bytes) in struct DiscResource
         +0x004e resources/disk.h: *59
dr_DiscSync      struct Interrupt(size 0x0016 bytes) in struct DiscResource
         +0x0064 resources/disk.h: *60
dr_Flags         unsigned char in struct DiscResource
         +0x0026 resources/disk.h: *53
dr_Index         struct Interrupt(size 0x0016 bytes) in struct DiscResource
         +0x007a resources/disk.h: *61
dr_Library       struct Library(size 0x0022 bytes) in struct DiscResource
         +0x0000 resources/disk.h: *51
dr_SysLib        pointer to struct Library in struct DiscResource
         +0x0028 resources/disk.h: *55
dr_UnitID        array [4] of unsigned long int in struct DiscResource
         +0x0030 resources/disk.h: *57
dr_Waiting       struct List(size 0x000e bytes) in struct DiscResource
         +0x0040 resources/disk.h: *58
dr_pad           unsigned char in struct DiscResource
         +0x0027 resources/disk.h: *54
dri_Depth        unsigned short int in struct DrawInfo
         +0x000c intuition/screens.h: *68
dri_Flags        unsigned long int in struct DrawInfo
         +0x0012 intuition/screens.h: *75
dri_Font         pointer to struct TextFont in struct DrawInfo
         +0x0008 intuition/screens.h: *67
dri_NumPens      unsigned short int in struct DrawInfo
         +0x0002 intuition/screens.h: *64
dri_Pens         pointer to unsigned short int in struct DrawInfo
         +0x0004 intuition/screens.h: *65
dri_Reserved     array [7] of unsigned long int in struct DrawInfo
         +0x0016 intuition/screens.h: *76
dri_Resolution   struct (no tag) (size 0x0004 bytes) in struct DrawInfo
         +0x000e intuition/screens.h: *73
dri_Version      unsigned short int in struct DrawInfo
         +0x0000 intuition/screens.h: *63
dru_DiscBlock    struct Interrupt(size 0x0016 bytes) in struct
                 DiscResourceUnit
         +0x0014 resources/disk.h: *45
dru_DiscSync     struct Interrupt(size 0x0016 bytes) in struct
                 DiscResourceUnit
         +0x002a resources/disk.h: *46
dru_Index        struct Interrupt(size 0x0016 bytes) in struct
                 DiscResourceUnit
         +0x0040 resources/disk.h: *47
dru_Message      struct Message(size 0x0014 bytes) in struct DiscResourceUnit
         +0x0000 resources/disk.h: *44
ds_Days          long int in struct DateStamp +0x0000 dos/dos.h: *55
ds_Minute        long int in struct DateStamp +0x0004 dos/dos.h: *56
ds_Tick          long int in struct DateStamp +0x0008 dos/dos.h: *57
```

Include File Cross Reference Page 33

```
dskbytr            unsigned short int in struct Custom
           +0x001a hardware/custom.h: *41
dskdat             unsigned short int in struct Custom
           +0x0026 hardware/custom.h: *46
dskdatr            unsigned short int in struct Custom
           +0x0008 hardware/custom.h: *32
dsklen             unsigned short int in struct Custom
           +0x0024 hardware/custom.h: *45
dskpt              pointer to void in struct Custom
           +0x0020 hardware/custom.h: *44
dsksync            unsigned short int in struct Custom
           +0x007e hardware/custom.h: *82
dummy              char in struct RastPort +0x001f graphics/rastport.h: *71
dvi_GlobVec        long int in struct DevInfo +0x0024 dos/dosextens.h: *355
dvi_Handler        long int in struct DevInfo +0x0010 dos/dosextens.h: *350
dvi_Lock           long int in struct DevInfo +0x000c dos/dosextens.h: *349
dvi_Name           long int in struct DevInfo +0x0028 dos/dosextens.h: *356
dvi_Next           long int in struct DevInfo +0x0000 dos/dosextens.h: *346
dvi_Priority       long int in struct DevInfo +0x0018 dos/dosextens.h: *352
dvi_SegList        long int in struct DevInfo +0x0020 dos/dosextens.h: *354
dvi_StackSize      long int in struct DevInfo +0x0014 dos/dosextens.h: *351
dvi_Startup        long int in struct DevInfo +0x001c dos/dosextens.h: *353
dvi_Task           pointer to void in struct DevInfo
           +0x0008 dos/dosextens.h: *348
dvi_Type           long int in struct DevInfo +0x0004 dos/dosextens.h: *347
dvp_DevNode        pointer to struct DosList in struct DevProc
           +0x000c dos/dosextens.h: *414
dvp_Flags          unsigned long int in struct DevProc
           +0x0008 dos/dosextens.h: *413
dvp_Lock           long int in struct DevProc +0x0004 dos/dosextens.h: *412
dvp_Port           pointer to struct MsgPort in struct DevProc
           +0x0000 dos/dosextens.h: *411
DAC_BINDTIME       #define 0x20 = 0x00000020  libraries/configregs.h: *264
DAC_BOOTTIME       #define 0x30 = 0x00000030  libraries/configregs.h: *260
DAC_BUSWIDTH       #define 0xC0 = 0x000000c0  libraries/configregs.h: *255
DAC_BYTEWIDE       #define 0x40 = 0x00000040  libraries/configregs.h: *257
DAC_CONFIGTIME     #define 0x10 = 0x00000010  libraries/configregs.h: *262
DAC_NEVER          #define 0x00 = 0x00000000  libraries/configregs.h: *261
DAC_NIBBLEWIDE     #define 0x00 = 0x00000000  libraries/configregs.h: *256
DAC_WORDWIDE       #define 0x80 = 0x00000080  libraries/configregs.h: *258
DBLPF              #define 0x400 = 0x00000400  graphics/display.h: *22
DBUFFER            #define 0x04 = 0x00000004  graphics/rastport.h: *103
DBufPacket         structure tag size 0x000c graphics/gels.h: *165, 235
DBuffer            pointer to struct DBufPacket in struct Bob
           +0x001a graphics/gels.h: *165
DDB_AllBit         #define 3 = 0x00000003  dos/dosasl.h: *113
DDB_Completed      #define 2 = 0x00000002  dos/dosasl.h: *111
DDB_ExaminedBit    #define 1 = 0x00000001  dos/dosasl.h: *109
DDB_PatternBit     #define 0 = 0x00000000  dos/dosasl.h: *107
DDB_Single         #define 4 = 0x00000004  dos/dosasl.h: *115
DDF_AllBit         #define 8 = 0x00000008  dos/dosasl.h: *114
DDF_Completed      #define 4 = 0x00000004  dos/dosasl.h: *112
DDF_ExaminedBit    #define 2 = 0x00000002  dos/dosasl.h: *110
DDF_PatternBit     #define 1 = 0x00000001  dos/dosasl.h: *108
DDF_Single         #define 16 = 0x00000010  dos/dosasl.h: *116
DEADEND_ALERT      #define 0x80000000 = 0x80000000  intuition/intuition.h: *1312
DEFARTIC           #define 100 = 0x00000064  devices/narrator.h: *70
DEFAULTMOUSEQUEUE #define (5) = 0x00000005  intuition/intuition.h: *960
DEFAULT_MONITOR_ID #define 0x00000000 = 0x00000000
                   graphics/displayinfo.h: *147
DEFAULT_MONITOR_NAME #define "default.monitor"  graphics/monitor.h: *65
DEFCENTRAL         #define 0 = 0x00000000  devices/narrator.h: *71
DEFERREFRESH       #define 0x8000 = 0x00008000  intuition/intuition.h: *206
DEFF0ENTHUS        #define 32 = 0x00000020  devices/narrator.h: *73
DEFF0PERT          #define 0 = 0x00000000  devices/narrator.h: *72
DEFFREQ            #define 22200 = 0x000056b8  devices/narrator.h: *62
```

Include File Cross Reference Page 34

```
DEFMODE            #define NATURALF0 = 0x00000000  devices/narrator.h: *69
DEFPITCH           #define 110 = 0x0000006e  devices/narrator.h: *59
DEFPRIORITY        #define 100 = 0x00000064  devices/narrator.h: *74
DEFRATE            #define 150 = 0x00000096  devices/narrator.h: *60
DEFSEX             #define MALE = 0x00000000  devices/narrator.h: *68
DEFVOL             #define 64 = 0x00000040  devices/narrator.h: *61
DELTAMOVE          #define IDCMP_DELTAMOVE = 0x00100000
                   intuition/iobsolete.h: *134
DEPTHIMAGE         #define (0x00L) = 0x00000000  intuition/imageclass.h: *101
DEST               #define 0x100 = 0x00000100  hardware/blit.h: *58
DESTADDR           #define u3.u4.u1.DestAddr  graphics/copper.h: *50
DESTDATA           #define u3.u4.u2.DestData  graphics/copper.h: *52
DETAILPEN          #define (0x0000) = 0x00000000  intuition/screens.h: *82
DEVICES_AUDIO_H    #define  devices/audio.h: *2
DEVICES_BOOTBLOCK_H #define  devices/bootblock.h: *2
DEVICES_CIA_H      #define 1 = 0x00000001  resources/cia.h: *2
DEVICES_CLIPBOARD_H #define  devices/clipboard.h: *2
                   libraries/iffparse.h: 25
DEVICES_CONSOLE_H #define  devices/console.h: *2
                   devices/conunit.h: 23
DEVICES_CONUNIT_H #define  devices/conunit.h: *2
DEVICES_GAMEPORT_H #define  devices/gameport.h: *2
DEVICES_HARDBLOCKS_H #define  devices/hardblocks.h: *2
DEVICES_INPUTEVENT_H #define  devices/inputevent.h: *2, 1
                   intuition/intuition.h: 47
DEVICES_INPUT_H    #define  devices/input.h: *2
DEVICES_KEYBOARD_H #define  devices/keyboard.h: *2
DEVICES_KEYMAP_H   #define  devices/keymap.h: *2, 1
DEVICES_NARRATOR_H #define  devices/narrator.h: *2
DEVICES_PARALLEL_H #define  devices/parallel.h: *2
                   devices/prtbase.h: 34
DEVICES_PRINTER_H #define  devices/printer.h: *2
DEVICES_PRTBASE_H #define  devices/prtbase.h: *2
DEVICES_PRTGFX_H   #define  devices/prtgfx.h: *2
DEVICES_SCSIDISK_H #define  devices/scsidisk.h: *2
DEVICES_SERIAL_H   #define  devices/serial.h: *2, 1
DEVICES_TIMER_H    #define 1 = 0x00000001  devices/timer.h: *2, 1
                   devices/prtbase.h: 40
                   dos/dosextens.h: 27
                   intuition/preferences.h: 19
DEVICES_TRACKDISK_H #define  devices/trackdisk.h: *2
DEV_ABORTIO        #define (-36) = 0xffffffdc  exec/io.h: *44
DEV_BEGINIO        #define (-30) = 0xffffffe2  exec/io.h: *43
DE_BAUD            #define 17 = 0x00000011  dos/filehandler.h: *78
DE_BLKSPERTRACK    #define 5 = 0x00000005  dos/filehandler.h: *62
DE_BOOTBLOCKS      #define 19 = 0x00000013  dos/filehandler.h: *80
DE_BOOTPRI         #define 15 = 0x0000000f  dos/filehandler.h: *74
DE_BUFMEMTYPE      #define 12 = 0x0000000c  dos/filehandler.h: *71
DE_CONTROL         #define 18 = 0x00000012  dos/filehandler.h: *79
DE_DOSTYPE         #define 16 = 0x00000010  dos/filehandler.h: *77
DE_INTERLEAVE      #define 8 = 0x00000008  dos/filehandler.h: *65
DE_LOWCYL          #define 9 = 0x00000009  dos/filehandler.h: *66
DE_MASK            #define 14 = 0x0000000e  dos/filehandler.h: *73
DE_MAXTRANSFER     #define 13 = 0x0000000d  dos/filehandler.h: *72
DE_MEMBUFTYPE      #define 12 = 0x0000000c  dos/filehandler.h: *69
DE_NUMBUFFERS      #define 11 = 0x0000000b  dos/filehandler.h: *68
DE_NUMHEADS        #define 3 = 0x00000003  dos/filehandler.h: *60
DE_PREFAC          #define 7 = 0x00000007  dos/filehandler.h: *64
DE_RESERVEDBLKS    #define 6 = 0x00000006  dos/filehandler.h: *63
DE_SECORG          #define 2 = 0x00000002  dos/filehandler.h: *59
DE_SECSPERBLK      #define 4 = 0x00000004  dos/filehandler.h: *61
DE_SIZEBLOCK       #define 1 = 0x00000001  dos/filehandler.h: *58
DE_TABLESIZE       #define 0 = 0x00000000  dos/filehandler.h: *57
DE_UPPERCYL        #define 10 = 0x0000000a  dos/filehandler.h: *67
DFH_ID             #define 0x0f80 = 0x00000f80  libraries/diskfont.h: *61
DFTCH_MASK         #define 0xFF = 0x000000ff  graphics/display.h: *37
```

```
DGB_REMOVABLE       #define 0 = 0x00000000  devices/trackdisk.h: *165
DGF_REMOVABLE       #define 1 = 0x00000001  devices/trackdisk.h: *166
DG_CDROM            #define 5 = 0x00000005  devices/trackdisk.h: *157
DG_COMMUNICATION #define 9 = 0x00000009  devices/trackdisk.h: *161
DG_DIRECT_ACCESS #define 0 = 0x00000000  devices/trackdisk.h: *152
DG_MEDIUM_CHANGER #define 8 = 0x00000008  devices/trackdisk.h: *160
DG_OPTICAL_DISK  #define 7 = 0x00000007  devices/trackdisk.h: *159
DG_PRINTER          #define 2 = 0x00000002  devices/trackdisk.h: *154
DG_PROCESSOR        #define 3 = 0x00000003  devices/trackdisk.h: *155
DG_SCANNER          #define 6 = 0x00000006  devices/trackdisk.h: *158
DG_SEQUENTIAL_ACCESS #define 1 = 0x00000001  devices/trackdisk.h: *153
DG_UNKNOWN          #define 31 = 0x0000001f  devices/trackdisk.h: *162
DG_WORM             #define 4 = 0x00000004  devices/trackdisk.h: *156
DHeight             short int in struct ViewPort +0x001a graphics/view.h: *50
DIAB_630            #define 0x04 = 0x00000004  intuition/preferences.h: *196
DIAB_ADV_D25        #define 0x05 = 0x00000005  intuition/preferences.h: *197
DIAB_C_150          #define 0x06 = 0x00000006  intuition/preferences.h: *198
DIMENSIONS_MASK  #define (BOUNDED_DIMENSIONS|ABSOLUTE_DIMENSIONS|
                            PIXEL_DIMENSIONS|MULTIPLY_DIMENSIONS) = 0x000000f0
                    intuition/preferences.h: *265
DIPF_IS_BEAMSYNC #define 0x00000800 = 0x00000800  graphics/displayinfo.h: *88
DIPF_IS_DRAGGABLE #define 0x00000200 = 0x00000200  graphics/displayinfo.h: *86
DIPF_IS_DUALPF      #define 0x00000002 = 0x00000002  graphics/displayinfo.h: *73
DIPF_IS_ECS         #define 0x00000010 = 0x00000010  graphics/displayinfo.h: *80
DIPF_IS_EXTRAHALFBRITE #define 0x00001000 = 0x00001000
                    graphics/displayinfo.h: *90
DIPF_IS_GENLOCK  #define 0x00000080 = 0x00000080  graphics/displayinfo.h: *83
DIPF_IS_HAM         #define 0x00000008 = 0x00000008  graphics/displayinfo.h: *75
DIPF_IS_LACE        #define 0x00000001 = 0x00000001  graphics/displayinfo.h: *72
DIPF_IS_PAL         #define 0x00000020 = 0x00000020  graphics/displayinfo.h: *81
DIPF_IS_PANELLED #define 0x00000400 = 0x00000400  graphics/displayinfo.h: *87
DIPF_IS_PF2PRI      #define 0x00000004 = 0x00000004  graphics/displayinfo.h: *74
DIPF_IS_SPRITES  #define 0x00000040 = 0x00000040  graphics/displayinfo.h: *82
DIPF_IS_WB          #define 0x00000100 = 0x00000100  graphics/displayinfo.h: *85
DISKINSERTED        #define IDCMP_DISKINSERTED = 0x00008000
                    intuition/iobsolete.h: *129
DISKNAME            #define "disk.resource"  resources/disk.h: *102
DISKREMOVED         #define IDCMP_DISKREMOVED = 0x00010000
                    intuition/iobsolete.h: *130
DISPLAYDUAL         #define 0x0040 = 0x00000040  hardware/custom.h: *153
DISPLAYNAMELEN      #define 32 = 0x00000020  graphics/displayinfo.h: *130, 135
DISPLAYPAL          #define 0x0020 = 0x00000020  hardware/custom.h: *154
DITHERING_MASK      #define (HALFTONE_DITHERING|FLOYD_DITHERING) = 0x00000600
                    intuition/preferences.h: *266
DIW_HORIZ_POS       #define 0x7F = 0x0000007f  graphics/display.h: *32
DIW_VRTCL_POS       #define 0x1FF = 0x000001ff  graphics/display.h: *33
DIW_VRTCL_POS_SHIFT #define 7 = 0x00000007  graphics/display.h: *34
DI_AVAIL_NOCHIPS #define 0x0001 = 0x00000001  graphics/displayinfo.h: *66
DI_AVAIL_NOMONITOR #define 0x0002 = 0x00000002  graphics/displayinfo.h: *67
DI_AVAIL_NOTWITHGENLOCK #define 0x0004 = 0x00000004
                    graphics/displayinfo.h: *68
DLT_DEVICE          #define 0 = 0x00000000  dos/dosextens.h: *402
DLT_DIRECTORY       #define 1 = 0x00000001  dos/dosextens.h: *403
DLT_LATE            #define 3 = 0x00000003  dos/dosextens.h: *405
DLT_NONBINDING      #define 4 = 0x00000004  dos/dosextens.h: *406
DLT_PRIVATE         #define -1 = 0xffffffff  dos/dosextens.h: *407
DLT_VOLUME          #define 2 = 0x00000002  dos/dosextens.h: *404
DMAB_AUD0           #define 0 = 0x00000000  hardware/dmabits.h: *37
DMAB_AUD1           #define 1 = 0x00000001  hardware/dmabits.h: *38
DMAB_AUD2           #define 2 = 0x00000002  hardware/dmabits.h: *39
DMAB_AUD3           #define 3 = 0x00000003  hardware/dmabits.h: *40
DMAB_BLITHOG        #define 10 = 0x0000000a  hardware/dmabits.h: *47
DMAB_BLITTER        #define 6 = 0x00000006  hardware/dmabits.h: *43
DMAB_BLTDONE        #define 14 = 0x0000000e  hardware/dmabits.h: *48
DMAB_BLTNZERO       #define 13 = 0x0000000d  hardware/dmabits.h: *49
DMAB_COPPER         #define 7 = 0x00000007  hardware/dmabits.h: *44
```

```
DMAB_DISK           #define 4 = 0x00000004  hardware/dmabits.h: *41
DMAB_MASTER         #define 9 = 0x00000009  hardware/dmabits.h: *46
DMAB_RASTER         #define 8 = 0x00000008  hardware/dmabits.h: *45
DMAB_SETCLR         #define 15 = 0x0000000f  hardware/dmabits.h: *36
DMAB_SPRITE         #define 5 = 0x00000005  hardware/dmabits.h: *42
DMAF_ALL            #define 0x01FF = 0x000001ff  hardware/dmabits.h: *29
DMAF_AUD0           #define 0x0001 = 0x00000001  hardware/dmabits.h: *18
DMAF_AUD1           #define 0x0002 = 0x00000002  hardware/dmabits.h: *19
DMAF_AUD2           #define 0x0004 = 0x00000004  hardware/dmabits.h: *20
DMAF_AUD3           #define 0x0008 = 0x00000008  hardware/dmabits.h: *21
DMAF_AUDIO          #define 0x000F = 0x0000000f  hardware/dmabits.h: *17
DMAF_BLITHOG        #define 0x0400 = 0x00000400  hardware/dmabits.h: *28
DMAF_BLITTER        #define 0x0040 = 0x00000040  hardware/dmabits.h: *24
DMAF_BLTDONE        #define 0x4000 = 0x00004000  hardware/dmabits.h: *33
DMAF_BLTNZERO       #define 0x2000 = 0x00002000  hardware/dmabits.h: *34
DMAF_COPPER         #define 0x0080 = 0x00000080  hardware/dmabits.h: *25
DMAF_DISK           #define 0x0010 = 0x00000010  hardware/dmabits.h: *22
DMAF_MASTER         #define 0x0200 = 0x00000200  hardware/dmabits.h: *27
DMAF_RASTER         #define 0x0100 = 0x00000100  hardware/dmabits.h: *26
DMAF_SETCLR         #define 0x8000 = 0x00008000  hardware/dmabits.h: *16
DMAF_SPRITE         #define 0x0020 = 0x00000020  hardware/dmabits.h: *23
DMODECOUNT          #define 0x0002 = 0x00000002  intuition/intuitionbase.h: *35
DMRequest           pointer to struct Requester in struct Window
            +0x0028 intuition/intuition.h: *815
DOSFALSE            #define (0L) = 0x00000000  dos/dos.h: *25
DOSNAME             #define "dos.library"  dos/dos.h: *20
DOSTRUE             #define (-1L) = 0xffffffff  dos/dos.h: *24
DOS_CLI             #define 4 = 0x00000004  dos/dos.h: *241
DOS_DATETIME_H      #define  dos/datetime.h: *2
DOS_DOSASL_H        #define  dos/dosasl.h: *2
DOS_DOSEXTENS_H     #define  dos/dosextens.h: *2, 1
DOS_DOSHUNKS_H      #define  dos/doshunks.h: *2
DOS_DOSTAGS_H       #define  dos/dostags.h: *2
DOS_DOS_H           #define  dos/dos.h: *2, 1
                    dos/filehandler.h: 19
                    dos/record.h: 17
                    dos/datetime.h: 17
                    dos/dosasl.h: 25
                    libraries/dos.h: 15
                    resources/filesysres.h: 21
DOS_EXALLCONTROL #define 1 = 0x00000001  dos/dos.h: *238
DOS_EXALL_H         #define  dos/exall.h: *2
DOS_FIB             #define 2 = 0x00000002  dos/dos.h: *239
DOS_FILEHANDLE      #define 0 = 0x00000000  dos/dos.h: *237
DOS_FILEHANDLER_H #define  dos/filehandler.h: *2
                    libraries/filehandler.h: 15
DOS_NOTIFY_H        #define  dos/notify.h: *2
DOS_RDARGS          #define 5 = 0x00000005  dos/dos.h: *242
DOS_RDARGS_H        #define  dos/rdargs.h: *2
DOS_RECORD_H        #define  dos/record.h: *2
DOS_STDIO_H         #define  dos/stdio.h: *2
DOS_STDPKT          #define 3 = 0x00000003  dos/dos.h: *240
DOS_VAR_H           #define  dos/var.h: *2
DOUBLE              typedef double  exec/types.h: *65
DOWNBACKGADGET      #define 1 = 0x00000001  intuition/intuitionbase.h: *48
DOWNIMAGE           #define (0x0DL) = 0x0000000d  intuition/imageclass.h: *109
DPB_DEAD            #define 3 = 0x00000003  devices/keymap.h: *70
DPB_MOD             #define 0 = 0x00000000  devices/keymap.h: *68
DPF_DEAD            #define 0x08 = 0x00000008  devices/keymap.h: *71
DPF_MOD             #define 0x01 = 0x00000001  devices/keymap.h: *69
DP_2DFACSHIFT       #define 4 = 0x00000004  devices/keymap.h: *74
DP_2DINDEXMASK      #define 0x0f = 0x0000000f  devices/keymap.h: *73
DRAFT               #define 0x000 = 0x00000000  intuition/preferences.h: *164
DRAGGADGET          #define 4 = 0x00000004  intuition/intuitionbase.h: *51
DRAWERDATAFILESIZE #define (sizeof(struct DrawerData)) = 0x0000003e
                    workbench/workbench.h: *60
```

Include File Cross Reference Page 37

```
DRB_ACTIVE       #define 7 = 0x00000007  resources/disk.h: *70
DRB_ALLOC0       #define 0 = 0x00000000  resources/disk.h: *66
DRB_ALLOC1       #define 1 = 0x00000001  resources/disk.h: *67
DRB_ALLOC2       #define 2 = 0x00000002  resources/disk.h: *68
DRB_ALLOC3       #define 3 = 0x00000003  resources/disk.h: *69
DRF_ACTIVE       #define (1<<7) = 0x00000080  resources/disk.h: *76
DRF_ALLOC0       #define (1<<0) = 0x00000001  resources/disk.h: *72
DRF_ALLOC1       #define (1<<1) = 0x00000002  resources/disk.h: *73
DRF_ALLOC2       #define (1<<2) = 0x00000004  resources/disk.h: *74
DRF_ALLOC3       #define (1<<3) = 0x00000008  resources/disk.h: *75
DRIF_NEWLOOK     #define 0x00000001 = 0x00000001  intuition/screens.h: *79
DRIVE3_5         #define 1 = 0x00000001  devices/trackdisk.h: *203
DRIVE3_5_150RPM  #define 3 = 0x00000003  devices/trackdisk.h: *205
DRIVE5_25        #define 2 = 0x00000002  devices/trackdisk.h: *204
DRI_VERSION      #define (1) = 0x00000001  intuition/screens.h: *59
DRT_150RPM       #define (0xAAAAAAAA) = 0xaaaaaaaa  resources/disk.h: *123
DRT_37422D2S     #define (0x55555555) = 0x55555555  resources/disk.h: *121
DRT_AMIGA        #define (0x00000000) = 0x00000000  resources/disk.h: *120
DRT_EMPTY        #define (0xFFFFFFFF) = 0xffffffff  resources/disk.h: *122
DR_ALLOCUNIT     #define (LIB_BASE - 0*LIB_VECTSIZE) = 0xfffffffa
                 resources/disk.h: *105
DR_FREEUNIT      #define (LIB_BASE - 1*LIB_VECTSIZE) = 0xfffffff4
                 resources/disk.h: *106
DR_GETUNIT       #define (LIB_BASE - 2*LIB_VECTSIZE) = 0xffffffee
                 resources/disk.h: *107
DR_GETUNITID     #define (LIB_BASE - 4*LIB_VECTSIZE) = 0xffffffe2
                 resources/disk.h: *109
DR_GIVEUNIT      #define (LIB_BASE - 3*LIB_VECTSIZE) = 0xffffffe8
                 resources/disk.h: *108
DR_LASTCOMM      #define (DR_READUNITID) = 0xffffffdc  resources/disk.h: *112
DR_READUNITID    #define (LIB_BASE - 5*LIB_VECTSIZE) = 0xffffffdc
                 resources/disk.h: *110
DSKDMAOFF        #define 0x4000 = 0x00004000  resources/disk.h: *87
DSR_CPR          #define 6 = 0x00000006  devices/console.h: *86
DTAG_DIMS        #define 0x80001000 = 0x80001000  graphics/displayinfo.h: *38
DTAG_DISP        #define 0x80000000 = 0x80000000  graphics/displayinfo.h: *37
DTAG_MNTR        #define 0x80002000 = 0x80002000  graphics/displayinfo.h: *39
DTAG_NAME        #define 0x80003000 = 0x80003000  graphics/displayinfo.h: *40
DTB_FUTURE       #define 1 = 0x00000001  dos/datetime.h: *43
DTB_SUBST        #define 0 = 0x00000000  dos/datetime.h: *41
DTF_FUTURE       #define 2 = 0x00000002  dos/datetime.h: *44
DTF_SUBST        #define 1 = 0x00000001  dos/datetime.h: *42
DT_DEV           #define 0L = 0x00000000  rexx/rexxio.h: *73
DT_DIR           #define 1L = 0x00000001  rexx/rexxio.h: *74
DT_VOL           #define 2L = 0x00000002  rexx/rexxio.h: *75
DUALPF           #define 0x0400 = 0x00000400  graphics/view.h: *96
DVPB_ASSIGN      #define 1 = 0x00000001  dos/dosextens.h: *420
DVPB_UNLOCK      #define 0 = 0x00000000  dos/dosextens.h: *418
DVPF_ASSIGN      #define (1L << DVPB_ASSIGN) = 0x00000002
                 dos/dosextens.h: *421
DVPF_UNLOCK      #define (1L << DVPB_UNLOCK) = 0x00000001
                 dos/dosextens.h: *419
DWidth           short int in struct ViewPort +0x0018 graphics/view.h: *50
DamageList       pointer to struct Region in struct Layer
         +0x009c graphics/clip.h: *60
DateStamp        structure tag size 0x000c dos/dos.h: *54, 72
                 dos/dosextens.h: 247, 336, 379
                 dos/datetime.h: 28
DateTime         structure tag size 0x001a dos/datetime.h: *27
Debug            char in struct GfxBase +0x00a1 graphics/gfxbase.h: *41
DebugData        pointer to void in struct ExecBase
         +0x0046 exec/execbase.h: *51
DebugEntry       pointer to void in struct ExecBase
         +0x0042 exec/execbase.h: *50
DefaultFont      pointer to struct TextFont in struct GfxBase
         +0x009a graphics/gfxbase.h: *38
```

Include File Cross Reference Page 38

```
DefaultTitle       pointer to unsigned char in struct Screen
           +0x001a intuition/screens.h: *111
DefaultTitle       pointer to unsigned char in struct NewScreen
           +0x0014 intuition/screens.h: *322
DefaultTitle       pointer to unsigned char in struct ExtNewScreen
           +0x0014 intuition/screens.h: *353
DeniseMaxDisplayColumn unsigned short int in struct MonitorSpec
           +0x0026 graphics/monitor.h: *35
DeniseMinDisplayColumn unsigned short int in struct MonitorSpec
           +0x0052 graphics/monitor.h: *48
Depth              unsigned char in struct BitMap +0x0005 graphics/gfx.h: *53
Depth              short int in struct Image +0x0008 intuition/intuition.h: *625
Depth              short int in struct NewScreen
           +0x0008 intuition/screens.h: *312
Depth              short int in struct ExtNewScreen
           +0x0008 intuition/screens.h: *348
Depth              short int in struct VSprite +0x001e graphics/gels.h: *105
Descendant         pointer to struct Window in struct Window
           +0x0046 intuition/intuition.h: *844
DestAddr           short int in union (no tag)  +0x0000 graphics/copper.h: *37
DestData           short int in union (no tag)  +0x0000 graphics/copper.h: *42
DetailPen          unsigned char in struct Window
           +0x0062 intuition/intuition.h: *859
DetailPen          unsigned char in struct NewWindow
           +0x0008 intuition/intuition.h: *979
DetailPen          unsigned char in struct ExtNewWindow
           +0x0008 intuition/intuition.h: *1049
DetailPen          unsigned char in struct Screen
           +0x014a intuition/screens.h: *137
DetailPen          unsigned char in struct NewScreen
           +0x000a intuition/screens.h: *314
DetailPen          unsigned char in struct ExtNewScreen
           +0x000a intuition/screens.h: *349
DetailPen          unsigned char in struct (no tag)
           +0x0000 intuition/cghooks.h: *53
DevInfo            structure tag size 0x002c dos/dosextens.h: *345
DevProc            structure tag size 0x0010 dos/dosextens.h: *410
Device             structure tag size 0x0022 exec/devices.h: *26
                   exec/io.h: 22, 31
                   devices/clipboard.h: 45
                   devices/printer.h: 144, 158
DeviceData         structure tag size 0x0034 devices/prtbase.h: *51, 66
DeviceList         struct List(size 0x000e bytes) in struct ExecBase
           +0x015e exec/execbase.h: *88
DeviceList         structure tag size 0x002c dos/dosextens.h: *331
DeviceNode         structure tag size 0x002c dos/filehandler.h: *101
DiagArea           structure tag size 0x000e libraries/configregs.h: *237
DimensionInfo      structure tag size 0x0058 graphics/displayinfo.h: *92
DiscResource       structure tag size 0x0094 resources/disk.h: *50
DiscResourceUnit   structure tag size 0x0056 resources/disk.h: *43, 52
DiskFontHeader     structure tag size 0x006a libraries/diskfont.h: *64
DiskObject         structure tag size 0x004e workbench/workbench.h: *62
DispCount          unsigned long int in struct ExecBase
           +0x011c exec/execbase.h: *66
DispCount          short int in struct StringInfo
           +0x0012 intuition/intuition.h: *536
DispPos            short int in struct StringInfo
           +0x000c intuition/intuition.h: *531
DisplayClip        struct Rectangle(size 0x0008 bytes) in struct ViewPortExtra
           +0x001c graphics/view.h: *83
DisplayCompatible unsigned long int in struct MonitorSpec
           +0x0054 graphics/monitor.h: *49
DisplayFlags       unsigned short int in struct GfxBase
           +0x00ce graphics/gfxbase.h: *53
DisplayID          unsigned long int in struct QueryHeader
           +0x0004 graphics/displayinfo.h: *45
```

```
                         Include File Cross Reference    Page 39
DisplayInfo       structure tag size 0x0030 graphics/displayinfo.h: *50
DisplayInfoDataBase struct List(size 0x000e bytes) in struct MonitorSpec
          +0x0058 graphics/monitor.h: *50
DisplayInfoDataBase pointer to void in struct GfxBase
          +0x0196 graphics/gfxbase.h: *89
DisplayInfoDataBaseSemaphore struct SignalSemaphore(size 0x002e bytes) in
                  struct MonitorSpec
          +0x0066 graphics/monitor.h: *51
DisplayInfoHandle "APTR"  graphics/displayinfo.h: *33
DosEnvec          structure tag size 0x0050 dos/filehandler.h: *29
DosInfo           structure tag size 0x0070 dos/dosextens.h: *274
DosLibrary        structure tag size 0x0042 dos/dosextens.h: *228
DosList           structure tag size 0x002c dos/dosextens.h: *361, 414
DosPacket         structure tag size 0x0030 dos/dosextens.h: *110, 143
DoubleClick       struct timeval(size 0x0008 bytes) in struct Preferences
          +0x0014 intuition/preferences.h: *58
DrawCircle        macro (4 arguments)   graphics/gfxmacros.h: *43
DrawInfo          structure tag (size 0x0032 bytes)in struct impDraw
                  intuition/screens.h: *61
                  intuition/cghooks.h: 61
                  intuition/imageclass.h: 143, 161
DrawMode          char in struct RastPort +0x001c graphics/rastport.h: *68
DrawMode          unsigned char in struct IntuiText
          +0x0002 intuition/intuition.h: *573
DrawMode          unsigned char in struct Border
          +0x0006 intuition/intuition.h: *602
DrawPath          pointer to struct VSprite in struct VSprite
          +0x0008 graphics/gels.h: *85
DrawerData        structure tag size 0x003e workbench/workbench.h: *52, 71
DriveGeometry     structure tag size 0x0020 devices/trackdisk.h: *137
DspIns            pointer to struct CopList in struct ViewPort
          +0x0008 graphics/view.h: *46
DxOffset          short int in struct ViewPort +0x001c graphics/view.h: *51
DxOffset          short int in struct View +0x000e graphics/view.h: *63
DyOffset          short int in struct CopList +0x0020 graphics/copper.h: *74
DyOffset          short int in struct ViewPort +0x001e graphics/view.h: *51
DyOffset          short int in struct View +0x000c graphics/view.h: *63
eac_Entries       unsigned long int in struct ExAllControl
          +0x0000 dos/exall.h: *64
eac_LastKey       unsigned long int in struct ExAllControl
          +0x0004 dos/exall.h: *65
eac_MatchFunc     pointer to struct Hook in struct ExAllControl
          +0x000c dos/exall.h: *67
eac_MatchString pointer to unsigned char in struct ExAllControl
          +0x0008 dos/exall.h: *66
eb_Private01      unsigned char in struct ExpansionBase
          +0x0023 libraries/expansionbase.h: *50
eb_Private02      unsigned long int in struct ExpansionBase
          +0x0024 libraries/expansionbase.h: *51
eb_Private03      unsigned long int in struct ExpansionBase
          +0x0028 libraries/expansionbase.h: *52
eb_Private04      struct CurrentBinding(size 0x0010 bytes) in struct
                  ExpansionBase
          +0x002c libraries/expansionbase.h: *53
eb_Private05      struct List(size 0x000e bytes) in struct ExpansionBase
          +0x003c libraries/expansionbase.h: *54
ec_BaseAddress    unsigned char in struct ExpansionControl
          +0x0002 libraries/configregs.h: *74
ec_Interrupt      unsigned char in struct ExpansionControl
          +0x0000 libraries/configregs.h: *72
ec_Reserved14     unsigned char in struct ExpansionControl
          +0x0004 libraries/configregs.h: *76
ec_Reserved15     unsigned char in struct ExpansionControl
          +0x0005 libraries/configregs.h: *77
ec_Reserved16     unsigned char in struct ExpansionControl
          +0x0006 libraries/configregs.h: *78
```

```
                         Include File Cross Reference    Page 40
ec_Reserved17     unsigned char in struct ExpansionControl
          +0x0007 libraries/configregs.h: *79
ec_Reserved18     unsigned char in struct ExpansionControl
          +0x0008 libraries/configregs.h: *80
ec_Reserved19     unsigned char in struct ExpansionControl
          +0x0009 libraries/configregs.h: *81
ec_Reserved1a     unsigned char in struct ExpansionControl
          +0x000a libraries/configregs.h: *82
ec_Reserved1b     unsigned char in struct ExpansionControl
          +0x000b libraries/configregs.h: *83
ec_Reserved1c     unsigned char in struct ExpansionControl
          +0x000c libraries/configregs.h: *84
ec_Reserved1d     unsigned char in struct ExpansionControl
          +0x000d libraries/configregs.h: *85
ec_Reserved1e     unsigned char in struct ExpansionControl
          +0x000e libraries/configregs.h: *86
ec_Reserved1f     unsigned char in struct ExpansionControl
          +0x000f libraries/configregs.h: *87
ec_Shutup         unsigned char in struct ExpansionControl
          +0x0003 libraries/configregs.h: *75
ec_Z3_HighBase    unsigned char in struct ExpansionControl
          +0x0001 libraries/configregs.h: *73
ed_Comment        pointer to unsigned char in struct ExAllData
          +0x0020 dos/exall.h: *49
ed_Days           unsigned long int in struct ExAllData +0x0014 dos/exall.h: *46
ed_Mins           unsigned long int in struct ExAllData +0x0018 dos/exall.h: *47
ed_Name           pointer to unsigned char in struct ExAllData
          +0x0004 dos/exall.h: *42
ed_Next           pointer to struct ExAllData in struct ExAllData
          +0x0000 dos/exall.h: *41
ed_Prot           unsigned long int in struct ExAllData +0x0010 dos/exall.h: *45
ed_Size           unsigned long int in struct ExAllData +0x000c dos/exall.h: *44
ed_Ticks          unsigned long int in struct ExAllData +0x001c dos/exall.h: *48
ed_Type           long int in struct ExAllData +0x0008 dos/exall.h: *43
er_Flags          unsigned char in struct ExpansionRom
          +0x0002 libraries/configregs.h: *51
er_InitDiagVec    unsigned short int in struct ExpansionRom
          +0x000a libraries/configregs.h: *55
er_Manufacturer unsigned short int in struct ExpansionRom
          +0x0004 libraries/configregs.h: *53
er_Product        unsigned char in struct ExpansionRom
          +0x0001 libraries/configregs.h: *50
er_Reserved03     unsigned char in struct ExpansionRom
          +0x0003 libraries/configregs.h: *52
er_Reserved0c     unsigned char in struct ExpansionRom
          +0x000c libraries/configregs.h: *56
er_Reserved0d     unsigned char in struct ExpansionRom
          +0x000d libraries/configregs.h: *57
er_Reserved0e     unsigned char in struct ExpansionRom
          +0x000e libraries/configregs.h: *58
er_Reserved0f     unsigned char in struct ExpansionRom
          +0x000f libraries/configregs.h: *59
er_SerialNumber unsigned long int in struct ExpansionRom
          +0x0006 libraries/configregs.h: *54
er_Type           unsigned char in struct ExpansionRom
          +0x0000 libraries/configregs.h: *49
es_Flags          unsigned long int in struct EasyStruct
          +0x0004 intuition/intuition.h: *1253
es_GadgetFormat pointer to unsigned char in struct EasyStruct
          +0x0010 intuition/intuition.h: *1256
es_StructSize     unsigned long int in struct EasyStruct
          +0x0000 intuition/intuition.h: *1252
es_TextFormat     pointer to unsigned char in struct EasyStruct
          +0x000c intuition/intuition.h: *1255
es_Title          pointer to unsigned char in struct EasyStruct
          +0x0008 intuition/intuition.h: *1254
```

Include File Cross Reference

```
estr_Nums          pointer to long int in struct ErrorString
           +0x0000 dos/dosextens.h: *219
estr_Strings       pointer to unsigned char in struct ErrorString
           +0x0004 dos/dosextens.h: *220
ev_hi              unsigned long int in struct EClockVal
           +0x0000 devices/timer.h: *33
ev_lo              unsigned long int in struct EClockVal
           +0x0004 devices/timer.h: *34
ex_CacheControl unsigned long int in struct ExecBase
           +0x023c exec/execbase.h: *137
ex_EClockFrequency unsigned long int in struct ExecBase
           +0x0238 exec/execbase.h: *136
ex_MMULock         pointer to void in struct ExecBase
           +0x0258 exec/execbase.h: *144
ex_Pad0            unsigned short int in struct ExecBase
           +0x022e exec/execbase.h: *128
ex_PoolThreshold unsigned long int in struct ExecBase
           +0x0248 exec/execbase.h: *141
ex_PublicPool      struct MinList(size 0x000c bytes) in struct ExecBase
           +0x024c exec/execbase.h: *142
ex_PuddleSize      unsigned long int in struct ExecBase
           +0x0244 exec/execbase.h: *140
ex_RamLibPrivate pointer to void in struct ExecBase
           +0x0234 exec/execbase.h: *130
ex_Reserved        array [12] of unsigned char in struct ExecBase
           +0x025c exec/execbase.h: *146
ex_Reserved0       unsigned long int in struct ExecBase
           +0x0230 exec/execbase.h: *129
ex_TaskID          unsigned long int in struct ExecBase
           +0x0240 exec/execbase.h: *138
exp                #define IEEEDPExp  libraries/mathffp.h: *45
                   libraries/mathieeedp.h: *45
ext_size           unsigned char in struct Preferences
           +0x00e7 intuition/preferences.h: *127
E                  #define ((float) 2.718281828459045)  libraries/mathffp.h: *22
                   libraries/mathieeedp.h: 22
EBB_BADMEM         #define 2 = 0x00000002  libraries/expansionbase.h: *72
EBB_CLOGGED        #define 0 = 0x00000000  libraries/expansionbase.h: *68
EBB_DOSFLAG        #define 3 = 0x00000003  libraries/expansionbase.h: *74
EBB_KICKBACK33     #define 4 = 0x00000004  libraries/expansionbase.h: *76
EBB_KICKBACK36     #define 5 = 0x00000005  libraries/expansionbase.h: *78
EBB_SHORTMEM       #define 1 = 0x00000001  libraries/expansionbase.h: *70
EBB_SILENTSTART    #define 6 = 0x00000006  libraries/expansionbase.h: *83
EBF_BADMEM         #define (1<<2) = 0x00000004  libraries/expansionbase.h: *73
EBF_CLOGGED        #define (1<<0) = 0x00000001  libraries/expansionbase.h: *69
EBF_DOSFLAG        #define (1<<3) = 0x00000008  libraries/expansionbase.h: *75
EBF_KICKBACK33     #define (1<<4) = 0x00000010  libraries/expansionbase.h: *77
EBF_KICKBACK36     #define (1<<5) = 0x00000020  libraries/expansionbase.h: *79
EBF_SHORTMEM       #define (1<<1) = 0x00000002  libraries/expansionbase.h: *71
EBF_SILENTSTART    #define (1<<6) = 0x00000040  libraries/expansionbase.h: *84
ECIB_INT2PEND      #define 4 = 0x00000004  libraries/configregs.h: *175
ECIB_INT6PEND      #define 5 = 0x00000005  libraries/configregs.h: *176
ECIB_INT7PEND      #define 6 = 0x00000006  libraries/configregs.h: *177
ECIB_INTENA        #define 1 = 0x00000001  libraries/configregs.h: *173
ECIB_INTERRUPTING #define 7 = 0x00000007  libraries/configregs.h: *178
ECIB_RESET         #define 3 = 0x00000003  libraries/configregs.h: *174
ECIF_INT2PEND      #define (1<<4) = 0x00000010  libraries/configregs.h: *182
ECIF_INT6PEND      #define (1<<5) = 0x00000020  libraries/configregs.h: *183
ECIF_INT7PEND      #define (1<<6) = 0x00000040  libraries/configregs.h: *184
ECIF_INTENA        #define (1<<1) = 0x00000002  libraries/configregs.h: *180
ECIF_INTERRUPTING #define (1<<7) = 0x00000080  libraries/configregs.h: *185
ECIF_RESET         #define (1<<3) = 0x00000008  libraries/configregs.h: *181
ECOFFSET           macro (1 argument)   libraries/configregs.h: *205
ECS_SPECIFIC       #define  graphics/view.h: *15
                   hardware/custom.h: 144
EC_MEMADDR         macro (1 argument)   libraries/configregs.h: *200
```

Include File Cross Reference

```
EClockVal          structure tag size 0x0008 devices/timer.h: *32
ED_COMMENT         #define 6 = 0x00000006  dos/exall.h: *33
ED_DATE            #define 5 = 0x00000005  dos/exall.h: *32
ED_NAME            #define 1 = 0x00000001  dos/exall.h: *28
ED_PROTECTION      #define 4 = 0x00000004  dos/exall.h: *31
ED_SIZE            #define 3 = 0x00000003  dos/exall.h: *30
ED_TYPE            #define 2 = 0x00000002  dos/exall.h: *29
EE_BADMEM          #define 44 = 0x0000002c  libraries/expansionbase.h: *65
EE_LASTBOARD       #define 40 = 0x00000028  libraries/expansionbase.h: *61
EE_NOBOARD         #define 43 = 0x0000002b  libraries/expansionbase.h: *64
EE_NOEXPANSION     #define 41 = 0x00000029  libraries/expansionbase.h: *62
EE_NOMEMORY        #define 42 = 0x0000002a  libraries/expansionbase.h: *63
EE_OK              #define 0 = 0x00000000  libraries/expansionbase.h: *60
EIGHT_LPI          #define 0x200 = 0x00000200  intuition/preferences.h: *169
ELITE              #define 0x400 = 0x00000400  intuition/preferences.h: *160
ENDGADGET          #define GACT_ENDGADGET = 0x00000004
                   intuition/iobsolete.h: *71
EO_BADFORMAT       #define (0x0009) = 0x00000009  intuition/sghooks.h: *81
EO_BIGCHANGE       #define (0x000A) = 0x0000000a  intuition/sghooks.h: *83
EO_CLEAR           #define (0x000C) = 0x0000000c  intuition/sghooks.h: *87
EO_DELBACKWARD     #define (0x0002) = 0x00000002  intuition/sghooks.h: *67
EO_DELFORWARD      #define (0x0003) = 0x00000003  intuition/sghooks.h: *69
EO_ENTER           #define (0x0005) = 0x00000005  intuition/sghooks.h: *73
EO_INSERTCHAR      #define (0x0008) = 0x00000008  intuition/sghooks.h: *79
EO_MOVECURSOR      #define (0x0004) = 0x00000004  intuition/sghooks.h: *71
EO_NOOP            #define (0x0001) = 0x00000001  intuition/sghooks.h: *65
EO_REPLACECHAR     #define (0x0007) = 0x00000007  intuition/sghooks.h: *77
EO_RESET           #define (0x0006) = 0x00000006  intuition/sghooks.h: *75
EO_SPECIAL         #define (0x000D) = 0x0000000d  intuition/sghooks.h: *89
EO_UNDO            #define (0x000B) = 0x0000000b  intuition/sghooks.h: *85
EPSON              #define 0x07 = 0x00000007  intuition/preferences.h: *199
EPSON_JX_80        #define 0x08 = 0x00000008  intuition/preferences.h: *200
ERFB_EXTENDED      #define 5 = 0x00000005  libraries/configregs.h: *160
ERFB_MEMSPACE      #define 7 = 0x00000007  libraries/configregs.h: *154
ERFB_NOSHUTUP      #define 6 = 0x00000006  libraries/configregs.h: *157
ERFB_ZORRO_III     #define 4 = 0x00000004  libraries/configregs.h: *164
ERFF_EXTENDED      #define (1<<5) = 0x00000020  libraries/configregs.h: *159
ERFF_MEMSPACE      #define (1<<7) = 0x00000080  libraries/configregs.h: *153
ERFF_NOSHUTUP      #define (1<<6) = 0x00000040  libraries/configregs.h: *156
ERFF_ZORRO_III     #define (1<<4) = 0x00000010  libraries/configregs.h: *163
EROFFSET           macro (1 argument)   libraries/configregs.h: *203
ERR10_001          #define (ERRC_MSG+1) = 0x00000001  rexx/errors.h: *16
ERR10_002          #define (ERRC_MSG+2) = 0x00000002  rexx/errors.h: *17
ERR10_003          #define (ERRC_MSG+3) = 0x00000003  rexx/errors.h: *18
ERR10_004          #define (ERRC_MSG+4) = 0x00000004  rexx/errors.h: *19
ERR10_005          #define (ERRC_MSG+5) = 0x00000005  rexx/errors.h: *20
ERR10_006          #define (ERRC_MSG+6) = 0x00000006  rexx/errors.h: *21
ERR10_007          #define (ERRC_MSG+7) = 0x00000007  rexx/errors.h: *22
ERR10_008          #define (ERRC_MSG+8) = 0x00000008  rexx/errors.h: *23
ERR10_009          #define (ERRC_MSG+9) = 0x00000009  rexx/errors.h: *24
ERR10_010          #define (ERRC_MSG+10) = 0x0000000a  rexx/errors.h: *26
ERR10_011          #define (ERRC_MSG+11) = 0x0000000b  rexx/errors.h: *27
ERR10_012          #define (ERRC_MSG+12) = 0x0000000c  rexx/errors.h: *28
ERR10_013          #define (ERRC_MSG+13) = 0x0000000d  rexx/errors.h: *29
ERR10_014          #define (ERRC_MSG+14) = 0x0000000e  rexx/errors.h: *30
ERR10_015          #define (ERRC_MSG+15) = 0x0000000f  rexx/errors.h: *31
ERR10_016          #define (ERRC_MSG+16) = 0x00000010  rexx/errors.h: *32
ERR10_017          #define (ERRC_MSG+17) = 0x00000011  rexx/errors.h: *33
ERR10_018          #define (ERRC_MSG+18) = 0x00000012  rexx/errors.h: *34
ERR10_019          #define (ERRC_MSG+19) = 0x00000013  rexx/errors.h: *35
ERR10_020          #define (ERRC_MSG+20) = 0x00000014  rexx/errors.h: *37
ERR10_021          #define (ERRC_MSG+21) = 0x00000015  rexx/errors.h: *38
ERR10_022          #define (ERRC_MSG+22) = 0x00000016  rexx/errors.h: *39
ERR10_023          #define (ERRC_MSG+23) = 0x00000017  rexx/errors.h: *40
ERR10_024          #define (ERRC_MSG+24) = 0x00000018  rexx/errors.h: *41
ERR10_025          #define (ERRC_MSG+25) = 0x00000019  rexx/errors.h: *42
```

```
ERR10_026        #define (ERRC_MSG+26) = 0x0000001a  rexx/errors.h: *43
ERR10_027        #define (ERRC_MSG+27) = 0x0000001b  rexx/errors.h: *44
ERR10_028        #define (ERRC_MSG+28) = 0x0000001c  rexx/errors.h: *45
ERR10_029        #define (ERRC_MSG+29) = 0x0000001d  rexx/errors.h: *46
ERR10_030        #define (ERRC_MSG+30) = 0x0000001e  rexx/errors.h: *48
ERR10_031        #define (ERRC_MSG+31) = 0x0000001f  rexx/errors.h: *49
ERR10_032        #define (ERRC_MSG+32) = 0x00000020  rexx/errors.h: *50
ERR10_033        #define (ERRC_MSG+33) = 0x00000021  rexx/errors.h: *51
ERR10_034        #define (ERRC_MSG+34) = 0x00000022  rexx/errors.h: *52
ERR10_035        #define (ERRC_MSG+35) = 0x00000023  rexx/errors.h: *53
ERR10_036        #define (ERRC_MSG+36) = 0x00000024  rexx/errors.h: *54
ERR10_037        #define (ERRC_MSG+37) = 0x00000025  rexx/errors.h: *55
ERR10_038        #define (ERRC_MSG+38) = 0x00000026  rexx/errors.h: *56
ERR10_039        #define (ERRC_MSG+39) = 0x00000027  rexx/errors.h: *57
ERR10_040        #define (ERRC_MSG+40) = 0x00000028  rexx/errors.h: *59
ERR10_041        #define (ERRC_MSG+41) = 0x00000029  rexx/errors.h: *60
ERR10_042        #define (ERRC_MSG+42) = 0x0000002a  rexx/errors.h: *61
ERR10_043        #define (ERRC_MSG+43) = 0x0000002b  rexx/errors.h: *62
ERR10_044        #define (ERRC_MSG+44) = 0x0000002c  rexx/errors.h: *63
ERR10_045        #define (ERRC_MSG+45) = 0x0000002d  rexx/errors.h: *64
ERR10_046        #define (ERRC_MSG+46) = 0x0000002e  rexx/errors.h: *65
ERR10_047        #define (ERRC_MSG+47) = 0x0000002f  rexx/errors.h: *66
ERR10_048        #define (ERRC_MSG+48) = 0x00000030  rexx/errors.h: *67
ERRC_MSG         #define 0 = 0x00000000  rexx/errors.h: *15
ERROR_ACTION_NOT_KNOWN #define 209 = 0x000000d1  dos/dos.h: *165
ERROR_BAD_HUNK   #define 235 = 0x000000eb  dos/dos.h: *187
ERROR_BAD_NUMBER #define 115 = 0x00000073  dos/dos.h: *150
ERROR_BAD_STREAM_NAME #define 206 = 0x000000ce  dos/dos.h: *163
ERROR_BAD_TEMPLATE #define 114 = 0x00000072  dos/dos.h: *149
ERROR_BREAK      #define 304 = 0x00000130  dos/dosasl.h: *149
ERROR_BUFFER_OVERFLOW #define 303 = 0x0000012f  dos/dosasl.h: *148
ERROR_COMMENT_TOO_BIG #define 220 = 0x000000dc  dos/dos.h: *176
ERROR_DELETE_PROTECTED #define 222 = 0x000000de  dos/dos.h: *178
ERROR_DEVICE_NOT_MOUNTED #define 218 = 0x000000da  dos/dos.h: *174
ERROR_DIRECTORY_NOT_EMPTY #define 216 = 0x000000d8  dos/dos.h: *172
ERROR_DIR_NOT_FOUND #define 204 = 0x000000cc  dos/dos.h: *161
ERROR_DISK_FULL  #define 221 = 0x000000dd  dos/dos.h: *177
ERROR_DISK_NOT_VALIDATED #define 213 = 0x000000d5  dos/dos.h: *169
ERROR_DISK_WRITE_PROTECTED #define 214 = 0x000000d6  dos/dos.h: *170
ERROR_FILE_NOT_OBJECT #define 121 = 0x00000079  dos/dos.h: *156
ERROR_INVALID_COMPONENT_NAME #define 210 = 0x000000d2  dos/dos.h: *166
ERROR_INVALID_LOCK #define 211 = 0x000000d3  dos/dos.h: *167
ERROR_INVALID_RESIDENT_LIBRARY #define 122 = 0x0000007a  dos/dos.h: *157
ERROR_IS_SOFT_LINK #define 233 = 0x000000e9  dos/dos.h: *185
ERROR_KEY_NEEDS_ARG #define 117 = 0x00000075  dos/dos.h: *152
ERROR_LINE_TOO_LONG #define 120 = 0x00000078  dos/dos.h: *155
ERROR_LOCK_COLLISION #define 241 = 0x000000f1  dos/dos.h: *190
ERROR_LOCK_TIMEOUT #define 242 = 0x000000f2  dos/dos.h: *191
ERROR_NOT_A_DOS_DISK #define 225 = 0x000000e1  dos/dos.h: *181
ERROR_NOT_EXECUTABLE #define 305 = 0x00000131  dos/dosasl.h: *150
ERROR_NOT_IMPLEMENTED #define 236 = 0x000000ec  dos/dos.h: *188
ERROR_NO_DEFAULT_DIR #define 201 = 0x000000c9  dos/dos.h: *158
ERROR_NO_DISK    #define 226 = 0x000000e2  dos/dos.h: *182
ERROR_NO_FREE_STORE #define 103 = 0x00000067  dos/dos.h: *147
ERROR_NO_MORE_ENTRIES #define 232 = 0x000000e8  dos/dos.h: *183
ERROR_OBJECT_EXISTS #define 203 = 0x000000cb  dos/dos.h: *160
ERROR_OBJECT_IN_USE #define 202 = 0x000000ca  dos/dos.h: *159
ERROR_OBJECT_LINKED #define 234 = 0x000000ea  dos/dos.h: *186
ERROR_OBJECT_NOT_FOUND #define 205 = 0x000000cd  dos/dos.h: *162
ERROR_OBJECT_TOO_LARGE #define 207 = 0x000000cf  dos/dos.h: *164
ERROR_OBJECT_WRONG_TYPE #define 212 = 0x000000d4  dos/dos.h: *168
ERROR_READ_PROTECTED #define 224 = 0x000000e0  dos/dos.h: *180
ERROR_RECORD_NOT_LOCKED #define 240 = 0x000000f0  dos/dos.h: *189
ERROR_RENAME_ACROSS_DEVICES #define 215 = 0x000000d7  dos/dos.h: *171
ERROR_REQUIRED_ARG_MISSING #define 116 = 0x00000074  dos/dos.h: *151
ERROR_SEEK_ERROR #define 219 = 0x000000db  dos/dos.h: *175
```

```
ERROR_TASK_TABLE_FULL #define 105 = 0x00000069  dos/dos.h: *148
ERROR_TOO_MANY_ARGS #define 118 = 0x00000076  dos/dos.h: *153
ERROR_TOO_MANY_LEVELS #define 217 = 0x000000d9  dos/dos.h: *173
ERROR_UNLOCK_ERROR #define 243 = 0x000000f3  dos/dos.h: *192
ERROR_UNMATCHED_QUOTES #define 119 = 0x00000077  dos/dos.h: *154
ERROR_WRITE_PROTECTED #define 223 = 0x000000df  dos/dos.h: *179
ERTB_CHAINEDCONFIG #define 3 = 0x00000003  libraries/configregs.h: *138
ERTB_DIAGVALID   #define 4 = 0x00000004  libraries/configregs.h: *137
ERTB_MEMLIST     #define 5 = 0x00000005  libraries/configregs.h: *136
ERTF_CHAINEDCONFIG #define (1<<3) = 0x00000008  libraries/configregs.h: *142
ERTF_DIAGVALID   #define (1<<4) = 0x00000010  libraries/configregs.h: *141
ERTF_MEMLIST     #define (1<<5) = 0x00000020  libraries/configregs.h: *140
ERT_MEMBIT       #define 0 = 0x00000000  libraries/configregs.h: *146
ERT_MEMMASK      #define 0x07 ;Bits 2-0  libraries/configregs.h: *145
ERT_MEMNEEDED    macro (1 argument)   libraries/configregs.h: *191
ERT_MEMSIZE      #define 3 = 0x00000003  libraries/configregs.h: *147
ERT_NEWBOARD     #define 0xc0 = 0x000000c0  libraries/configregs.h: *131
ERT_SLOTSNEEDED macro (1 argument)   libraries/configregs.h: *195
ERT_TYPEBIT      #define 6 = 0x00000006  libraries/configregs.h: *129
ERT_TYPEMASK     #define 0xc0 ;Bits 7-6  libraries/configregs.h: *128
ERT_TYPESIZE     #define 2 = 0x00000002  libraries/configregs.h: *130
ERT_Z3_SSBIT     #define 0 = 0x00000000  libraries/configregs.h: *167
ERT_Z3_SSMASK    #define 0x0F = 0x0000000f  libraries/configregs.h: *166
ERT_Z3_SSSIZE    #define 4 = 0x00000004  libraries/configregs.h: *168
ERT_ZORROII      #define ERT_NEWBOARD = 0x000000c0
                 libraries/configregs.h: *132
ERT_ZORROIII     #define 0x80 = 0x00000080  libraries/configregs.h: *133
ETD_CLEAR        #define (CMD_CLEAR|TDF_EXTCOM) = 0x00008005
                 devices/trackdisk.h: *109
ETD_FORMAT       #define (TD_FORMAT|TDF_EXTCOM) = 0x0000800b
                 devices/trackdisk.h: *107
ETD_MOTOR        #define (TD_MOTOR|TDF_EXTCOM) = 0x00008009
                 devices/trackdisk.h: *105
ETD_RAWREAD      #define (TD_RAWREAD|TDF_EXTCOM) = 0x00008010
                 devices/trackdisk.h: *110
ETD_RAWWRITE     #define (TD_RAWWRITE|TDF_EXTCOM) = 0x00008011
                 devices/trackdisk.h: *111
ETD_READ         #define (CMD_READ|TDF_EXTCOM) = 0x00008002
                 devices/trackdisk.h: *104
ETD_SEEK         #define (TD_SEEK|TDF_EXTCOM) = 0x0000800a
                 devices/trackdisk.h: *106
ETD_UPDATE       #define (CMD_UPDATE|TDF_EXTCOM) = 0x00008004
                 devices/trackdisk.h: *108
ETD_WRITE        #define (CMD_WRITE|TDF_EXTCOM) = 0x00008003
                 devices/trackdisk.h: *103
EVENTMAX         #define 10 = 0x0000000a  intuition/intuitionbase.h: *39
EXAMINE_BIT      #define 2 = 0x00000002  dos/dosasl.h: *140
EXCLUSIVE_LOCK   #define -1 = 0xffffffff  dos/dos.h: *51
EXEC_ALERTS_H    #define  exec/alerts.h: *2, 1
EXEC_DEVICES_H   #define  exec/devices.h: *2, 1
                 devices/trackdisk.h: 22
EXEC_ERRORS_H    #define  exec/errors.h: *2, 1
EXEC_EXECBASE_H  #define  exec/execbase.h: *2, 1
EXEC_EXEC_H      #define  exec/exec.h: *2
EXEC_INITIALIZERS_H #define  exec/initializers.h: *2, 1
EXEC_INTERRUPTS_H #define  exec/interrupts.h: *2, 1
                 exec/execbase.h: 19
                 graphics/gfxbase.h: 21
                 intuition/intuitionbase.h: 28
                 resources/disk.h: 27
EXEC_IO_H        #define  exec/io.h: *2, 1(2)
                 devices/audio.h: 15
                 devices/console.h: 19
                 devices/gameport.h: 19
                 devices/input.h: 15
                 devices/keyboard.h: 15
```

Include File Cross Reference

Page 45

```
                  devices/narrator.h: 17
                  devices/parallel.h: 15
                  devices/serial.h: 15
                  devices/trackdisk.h: 18
EXEC_LIBRARIES_H  #define  exec/libraries.h: *2, 1(2)
                  exec/execbase.h: 23
                  devices/prtbase.h: 27
                  dos/dosextens.h: 21
                  dos/dosasl.h: 17
                  graphics/gfxbase.h: 18
                  intuition/intuitionbase.h: 19
                  libraries/asl.h: 24
                  libraries/expansionbase.h: 19
                  libraries/mathlibrary.h: 17
                  resources/disk.h: 31
                  resources/misc.h: 19
                  rexx/storage.h: 31
EXEC_LISTS_H      #define  exec/lists.h: *2, 1(2)
                  exec/tasks.h: 19
                  exec/interrupts.h: 19
                  exec/semaphores.h: 19
                  exec/execbase.h: 15
                  devices/clipboard.h: 21
                  devices/keymap.h: 19
                  devices/printer.h: 23
                  devices/prtbase.h: 21
                  graphics/layers.h: 15
                  dos/dosasl.h: 21
                  graphics/gfxbase.h: 15
                  libraries/asl.h: 20
                  libraries/diskfont.h: 21
                  libraries/iffparse.h: 19
                  resources/disk.h: 19
                  resources/filesysres.h: 18
                  rexx/storage.h: 23
                  workbench/workbench.h: 23
EXEC_MEMORY_H     #define  exec/memory.h: *2, 1
EXEC_NODES_H      #define  exec/nodes.h: *2, 1(2)
                  exec/ports.h: 15
                  exec/lists.h: 15
                  exec/tasks.h: 15
                  exec/memory.h: 15
                  exec/interrupts.h: 15
                  exec/semaphores.h: 15
                  devices/clipboard.h: 18
                  devices/keymap.h: 16
                  devices/printer.h: 19
                  devices/prtbase.h: 18
                  utility/hooks.h: 19
                  graphics/gfxnodes.h: 15
                  dos/rdargs.h: 21
                  dos/var.h: 18
                  graphics/graphint.h: 15
                  libraries/configvars.h: 19
                  libraries/diskfont.h: 18
                  libraries/mathresource.h: 17
                  resources/filesysres.h: 15
                  rexx/storage.h: 19
                  workbench/workbench.h: 19
EXEC_PORTS_H      #define  exec/ports.h: *2, 1(2)
                  exec/semaphores.h: 23
                  exec/io.h: 15
                  devices/clipboard.h: 24
                  devices/conunit.h: 19
                  devices/printer.h: 27
                  devices/prtbase.h: 24
```

Include File Cross Reference

Page 46

```
                   dos/dosextens.h: 18
                   graphics/text.h: 15
                   intuition/intuition.h: 43
                   dos/filehandler.h: 15
                   dos/notify.h: 21
                   workbench/startup.h: 19
                   libraries/iffparse.h: 22
                   resources/disk.h: 23
                   rexx/storage.h: 27
EXEC_RESIDENT_H    #define  exec/resident.h: *2, 1
EXEC_SEMAPHORES_H #define  exec/semaphores.h: *2, 1
                   dos/dosextens.h: 24
                   graphics/clip.h: 22
                   graphics/monitor.h: 15
                   graphics/layers.h: 19
                   libraries/expansionbase.h: 23
EXEC_TASKS_H       #define  exec/tasks.h: *2, 1(2)
                   exec/semaphores.h: 27
                   exec/execbase.h: 27
                   devices/prtbase.h: 30
                   dos/dosextens.h: 15
                   dos/notify.h: 25
                   workbench/workbench.h: 27
EXEC_TYPES_H       #define  exec/types.h: *2
                   exec/resident.h: 15
                   devices/prtgfx.h: 15
                   devices/scsidisk.h: 16
                   devices/bootblock.h: 15
                   devices/clipboard.h: 15
                   devices/console.h: 15
                   devices/conunit.h: 15
                   devices/gameport.h: 15
                   devices/hardblocks.h: 15
                   devices/printer.h: 15
                   devices/prtbase.h: 15
                   dos/dos.h: 15
                   intuition/intuition.h: 15
                   graphics/gfx.h: 15
                   graphics/clip.h: 15
                   utility/hooks.h: 15
                   graphics/view.h: 17
                   graphics/copper.h: 15
                   hardware/custom.h: 15
                   graphics/rastport.h: 15
                   utility/tagitem.h: 15
                   intuition/screens.h: 15
                   intuition/preferences.h: 15
                   dos/exall.h: 17
                   dos/notify.h: 17
                   dos/rdargs.h: 17
                   graphics/displayinfo.h: 15
                   graphics/gels.h: 15
                   graphics/gfxmacros.h: 15
                   graphics/regions.h: 15
                   graphics/scale.h: 15
                   graphics/sprite.h: 15
                   graphics/videocontrol.h: 15
                   hardware/cia.h: 16
                   intuition/cghooks.h: 15
                   intuition/gadgetclass.h: 15
                   intuition/intuitionbase.h: 15
                   intuition/sghooks.h: 15
                   libraries/asl.h: 16
                   workbench/startup.h: 15
                   libraries/commodities.h: 16
                   libraries/configregs.h: 16
```

```
                    libraries/configvars.h: 15
                    libraries/diskfont.h: 15
                    libraries/expansionbase.h: 15
                    libraries/gadtools.h: 17
                    libraries/iffparse.h: 16
                    resources/disk.h: 15
                    resources/misc.h: 15
                    rexx/storage.h: 15
                    workbench/workbench.h: 15
EXPANSIONNAME       #define "expansion.library"  libraries/expansion.h: *15
EXTENDED_MODE       #define 0x1000 = 0x00001000  graphics/view.h: *98
EXTEND_VSTRUCT      #define 0x1000 = 0x00001000  graphics/view.h: *86
EXTRAHALFBRITELACE_KEY #define 0x00000084 = 0x00000084
                    graphics/displayinfo.h: *176
EXTRAHALFBRITE_KEY #define 0x00000080 = 0x00000080
                    graphics/displayinfo.h: *175
EXTRA_HALFBRITE     #define 0x0080 = 0x00000080  graphics/view.h: *94
EXT_ABS             #define 2 = 0x00000002  dos/doshunks.h: *43
EXT_COMMON          #define 130 = 0x00000082  dos/doshunks.h: *46
EXT_DEF             #define 1 = 0x00000001  dos/doshunks.h: *42
EXT_DEXT16          #define 134 = 0x00000086  dos/doshunks.h: *50
EXT_DEXT32          #define 133 = 0x00000085  dos/doshunks.h: *49
EXT_DEXT8           #define 135 = 0x00000087  dos/doshunks.h: *51
EXT_REF16           #define 131 = 0x00000083  dos/doshunks.h: *47
EXT_REF32           #define 129 = 0x00000081  dos/doshunks.h: *45
EXT_REF8            #define 132 = 0x00000084  dos/doshunks.h: *48
EXT_RES             #define 3 = 0x00000003  dos/doshunks.h: *44
EXT_SYMB            #define 0 = 0x00000000  dos/doshunks.h: *41
EZ3_CONFIGAREA      #define 0x40000000 = 0x40000000  libraries/configregs.h: *119
EZ3_CONFIGAREAEND #define 0x7FFFFFFF = 0x7fffffff
                    libraries/configregs.h: *120
EZ3_EXPANSIONBASE #define 0xff000000 = 0xff000000
                    libraries/configregs.h: *110
EZ3_SIZEGRANULARITY #define 0x00080000 = 0x00080000
                    libraries/configregs.h: *121
E_EXPANSIONBASE     #define 0x00e80000 = 0x00e80000  libraries/configregs.h: *109
E_EXPANSIONSIZE     #define 0x00080000 = 0x00080000  libraries/configregs.h: *112
E_EXPANSIONSLOTS #define 8 = 0x00000008  libraries/configregs.h: *113
E_MEMORYBASE        #define 0x00200000 = 0x00200000  libraries/configregs.h: *115
E_MEMORYSIZE        #define 0x00800000 = 0x00800000  libraries/configregs.h: *116
E_MEMORYSLOTS       #define 128 = 0x00000080  libraries/configregs.h: *117
E_SLOTMASK          #define 0xffff = 0x0000ffff  libraries/configregs.h: *103
E_SLOTSHIFT         #define 16 = 0x00000010  libraries/configregs.h: *104
E_SLOTSIZE          #define 0x10000 = 0x00010000  libraries/configregs.h: *102
EasyStruct          structure tag size 0x0014 intuition/intuition.h: *1251
EditHook            pointer to struct Hook in struct StringExtend
             +0x000c intuition/sghooks.h: *27
EditOp              unsigned short int in struct SGWork
             +0x002a intuition/sghooks.h: *50
Elapsed             unsigned short int in struct ExecBase
             +0x0122 exec/execbase.h: *68
EnableCLI           short int in struct Preferences
             +0x007c intuition/preferences.h: *80
ErrorString         structure tag size 0x0008 dos/dosextens.h: *218, 235
ExAllControl        structure tag size 0x0010 dos/exall.h: *63
ExAllData           structure tag size 0x0024 dos/exall.h: *40, 41
ExecBase            structure tag size 0x0268 exec/execbase.h: *36
ExecBase            pointer to unsigned long int in struct GfxBase
             +0x01a2 graphics/gfxbase.h: *92
ExecMessage         struct Message(size 0x0014 bytes) in struct IntuiMessage
             +0x0000 intuition/intuition.h: *679
ExpansionBase       structure tag size 0x0058 libraries/expansionbase.h: *46
ExpansionControl structure tag size 0x0010 libraries/configregs.h: *71
ExpansionRom        structure tag size 0x0010 libraries/configregs.h: *48
                    libraries/configvars.h: 38
ExtData             pointer to unsigned char in struct Window
```

```
             +0x0074 intuition/intuition.h: *883
ExtData             pointer to unsigned char in struct Screen
             +0x0152 intuition/screens.h: *147
ExtNewScreen        structure tag size 0x0024 intuition/screens.h: *346
ExtNewWindow        structure tag size 0x0034 intuition/intuition.h: *1044
ExtendedModes       unsigned char in struct ViewPort +0x0023 graphics/view.h: *54
ExtendedNode        structure tag size 0x0018 graphics/gfxnodes.h: *19
                    graphics/monitor.h: 29, 144
                    graphics/view.h: 72, 81
Extension           pointer to struct StringExtend in struct StringInfo
             +0x0018 intuition/intuition.h: *546
Extension           pointer to struct TagItem in struct ExtNewWindow
             +0x0030 intuition/intuition.h: *1082
Extension           pointer to struct TagItem in struct ExtNewScreen
             +0x0020 intuition/screens.h: *357
fabs                #define IEEEDPAbs  libraries/mathffp.h: *35
                    libraries/mathieeedp.h: *35
fatten_count        char in struct Layer_Info +0x005a graphics/layers.h: *44
fc_FileName         array [256] of char in struct FontContents
             +0x0000 libraries/diskfont.h: *31
fc_Flags            unsigned char in struct FontContents
             +0x0103 libraries/diskfont.h: *34
fc_Style            unsigned char in struct FontContents
             +0x0102 libraries/diskfont.h: *33
fc_YSize            unsigned short int in struct FontContents
             +0x0100 libraries/diskfont.h: *32
fch_FileID          unsigned short int in struct FontContentsHeader
             +0x0000 libraries/diskfont.h: *55
fch_NumEntries      unsigned short int in struct FontContentsHeader
             +0x0002 libraries/diskfont.h: *56
fh_Arg1             #define fh_Args  dos/dosextens.h: *104
fh_Arg2             long int in struct FileHandle +0x0028 dos/dosextens.h: *105
fh_Args             long int in struct FileHandle +0x0024 dos/dosextens.h: *103
fh_Buf              long int in struct FileHandle +0x000c dos/dosextens.h: *96
fh_End              long int in struct FileHandle +0x0014 dos/dosextens.h: *98
fh_Func1            #define fh_Funcs  dos/dosextens.h: *100
fh_Func2            long int in struct FileHandle +0x001c dos/dosextens.h: *101
fh_Func3            long int in struct FileHandle +0x0020 dos/dosextens.h: *102
fh_Funcs            long int in struct FileHandle +0x0018 dos/dosextens.h: *99
fh_Link             pointer to struct Message in struct FileHandle
             +0x0000 dos/dosextens.h: *92
fh_Port             pointer to struct MsgPort in struct FileHandle
             +0x0004 dos/dosextens.h: *93
fh_Pos              long int in struct FileHandle +0x0010 dos/dosextens.h: *97
fh_Type             pointer to struct MsgPort in struct FileHandle
             +0x0008 dos/dosextens.h: *94
fhb_ChkSum          long int in struct FileSysHeaderBlock
             +0x0008 devices/hardblocks.h: *159
fhb_DosType         unsigned long int in struct FileSysHeaderBlock
             +0x0020 devices/hardblocks.h: *164
fhb_Flags           unsigned long int in struct FileSysHeaderBlock
             +0x0014 devices/hardblocks.h: *162
fhb_GlobalVec       long int in struct FileSysHeaderBlock
             +0x004c devices/hardblocks.h: *181
fhb_Handler         unsigned long int in struct FileSysHeaderBlock
             +0x0038 devices/hardblocks.h: *174
fhb_HostID          unsigned long int in struct FileSysHeaderBlock
             +0x000c devices/hardblocks.h: *160
fhb_ID              unsigned long int in struct FileSysHeaderBlock
             +0x0000 devices/hardblocks.h: *157
fhb_Lock            unsigned long int in struct FileSysHeaderBlock
             +0x0034 devices/hardblocks.h: *173
fhb_Next            unsigned long int in struct FileSysHeaderBlock
             +0x0010 devices/hardblocks.h: *161
fhb_PatchFlags      unsigned long int in struct FileSysHeaderBlock
             +0x0028 devices/hardblocks.h: *167
```

Include File Cross Reference Page 49

```
fhb_Priority     long int in struct FileSysHeaderBlock
         +0x0040 devices/hardblocks.h: *176
fhb_Reserved1    array [2] of unsigned long int in struct FileSysHeaderBlock
         +0x0018 devices/hardblocks.h: *163
fhb_Reserved2    array [23] of unsigned long int in struct FileSysHeaderBlock
         +0x0050 devices/hardblocks.h: *182
fhb_Reserved3    array [21] of unsigned long int in struct FileSysHeaderBlock
         +0x00ac devices/hardblocks.h: *183
fhb_SegListBlocks long int in struct FileSysHeaderBlock
         +0x0048 devices/hardblocks.h: *178
fhb_StackSize    unsigned long int in struct FileSysHeaderBlock
         +0x003c devices/hardblocks.h: *175
fhb_Startup      long int in struct FileSysHeaderBlock
         +0x0044 devices/hardblocks.h: *177
fhb_SummedLongs unsigned long int in struct FileSysHeaderBlock
         +0x0004 devices/hardblocks.h: *158
fhb_Task         unsigned long int in struct FileSysHeaderBlock
         +0x0030 devices/hardblocks.h: *172
fhb_Type         unsigned long int in struct FileSysHeaderBlock
         +0x002c devices/hardblocks.h: *171
fhb_Version      unsigned long int in struct FileSysHeaderBlock
         +0x0024 devices/hardblocks.h: *166
fib_Comment      array [80] of char in struct FileInfoBlock
         +0x0090 dos/dos.h: *73
fib_Date         struct DateStamp(size 0x000c bytes) in struct FileInfoBlock
         +0x0084 dos/dos.h: *72
fib_DirEntryType long int in struct FileInfoBlock +0x0004 dos/dos.h: *65
fib_DiskKey      long int in struct FileInfoBlock +0x0000 dos/dos.h: *64
fib_EntryType    long int in struct FileInfoBlock +0x0078 dos/dos.h: *69
fib_FileName     array [108] of char in struct FileInfoBlock
         +0x0008 dos/dos.h: *67
fib_NumBlocks    long int in struct FileInfoBlock +0x0080 dos/dos.h: *71
fib_Protection   long int in struct FileInfoBlock +0x0074 dos/dos.h: *68
fib_Reserved     array [36] of char in struct FileInfoBlock
         +0x00e0 dos/dos.h: *74
fib_Size         long int in struct FileInfoBlock +0x007c dos/dos.h: *70
firstBlissObj    pointer to void in struct GelsInfo
         +0x001e graphics/rastport.h: *53
fl_Access        long int in struct FileLock +0x0008 dos/dosextens.h: *446
fl_Key           long int in struct FileLock +0x0004 dos/dosextens.h: *445
fl_Link          long int in struct FileLock +0x0000 dos/dosextens.h: *444
fl_MemList       struct List(size 0x000e bytes) in struct FreeList
         +0x0002 workbench/workbench.h: *85
fl_NumFree       short int in struct FreeList
         +0x0000 workbench/workbench.h: *84
fl_Task          pointer to struct MsgPort in struct FileLock
         +0x000c dos/dosextens.h: *447
fl_Volume        long int in struct FileLock +0x0010 dos/dosextens.h: *448
flags            unsigned char in struct narrator_rb
         +0x0045 devices/narrator.h: *106
floor            #define IEEEDPFloor  libraries/mathffp.h: *36
                 libraries/mathieeedp.h: *36
fo_Attr          struct TextAttr(size 0x0008 bytes) in struct FontRequester
         +0x0008 libraries/asl.h: *148
fo_BackPen       unsigned char in struct FontRequester
         +0x0011 libraries/asl.h: *150
fo_DrawMode      unsigned char in struct FontRequester
         +0x0012 libraries/asl.h: *151
fo_FrontPen      unsigned char in struct FontRequester
         +0x0010 libraries/asl.h: *149
fo_Reserved1     array [2] of pointer to void in struct FontRequester
         +0x0000 libraries/asl.h: *147
fo_UserData      pointer to void in struct FontRequester
         +0x0014 libraries/asl.h: *152
front            pointer to struct Layer in struct Layer
         +0x0000 graphics/clip.h: *36
```

Include File Cross Reference Page 50

```
fse_DosType      unsigned long int in struct FileSysEntry
         +0x000e resources/filesysres.h: *36
fse_GlobalVec    long int in struct FileSysEntry
         +0x003a resources/filesysres.h: *50
fse_Handler      long int in struct FileSysEntry
         +0x0026 resources/filesysres.h: *45
fse_Lock         long int in struct FileSysEntry
         +0x0022 resources/filesysres.h: *44
fse_Node         struct Node(size 0x000e bytes) in struct FileSysEntry
         +0x0000 resources/filesysres.h: *34
fse_PatchFlags   unsigned long int in struct FileSysEntry
         +0x0016 resources/filesysres.h: *38
fse_Priority     long int in struct FileSysEntry
         +0x002e resources/filesysres.h: *47
fse_SegList      long int in struct FileSysEntry
         +0x0036 resources/filesysres.h: *49
fse_StackSize    unsigned long int in struct FileSysEntry
         +0x002a resources/filesysres.h: *46
fse_Startup      long int in struct FileSysEntry
         +0x0032 resources/filesysres.h: *48
fse_Task         unsigned long int in struct FileSysEntry
         +0x001e resources/filesysres.h: *43
fse_Type         unsigned long int in struct FileSysEntry
         +0x001a resources/filesysres.h: *42
fse_Version      unsigned long int in struct FileSysEntry
         +0x0012 resources/filesysres.h: *37
fsr_Creator      pointer to char in struct FileSysResource
         +0x000e resources/filesysres.h: *29
fsr_FileSysEntries struct List(size 0x000e bytes) in struct FileSysResource
         +0x0012 resources/filesysres.h: *30
fsr_Node         struct Node(size 0x000e bytes) in struct FileSysResource
         +0x0000 resources/filesysres.h: *28
fssm_Device      long int in struct FileSysStartupMsg
         +0x0004 dos/filehandler.h: *88
fssm_Environ     long int in struct FileSysStartupMsg
         +0x0008 dos/filehandler.h: *89
fssm_Flags       unsigned long int in struct FileSysStartupMsg
         +0x000c dos/filehandler.h: *90
fssm_Unit        unsigned long int in struct FileSysStartupMsg
         +0x0000 dos/filehandler.h: *87
function         pointer to function returning int in struct bltnode
         +0x0004 hardware/blit.h: *93
F0enthusiasm     unsigned char in struct narrator_rb
         +0x0046 devices/narrator.h: *107
F0perturb        unsigned char in struct narrator_rb
         +0x0047 devices/narrator.h: *108
F1adj            char in struct narrator_rb +0x0048 devices/narrator.h: *109
F2adj            char in struct narrator_rb +0x0049 devices/narrator.h: *110
F3adj            char in struct narrator_rb +0x004a devices/narrator.h: *111
FALSE            #define 0 = 0x00000000  exec/types.h: *73
FANFOLD          #define 0x00 = 0x00000000  intuition/preferences.h: *155
FAULT_MAX        #define 82 = 0x00000052  dos/dos.h: *98
FCH_ID           #define 0x0f00 = 0x00000f00  libraries/diskfont.h: *51
FEMALE           #define 1 = 0x00000001  devices/narrator.h: *64
FIBB_ARCHIVE     #define 4 = 0x00000004  dos/dos.h: *82
FIBB_DELETE      #define 0 = 0x00000000  dos/dos.h: *86
FIBB_EXECUTE     #define 1 = 0x00000001  dos/dos.h: *85
FIBB_PURE        #define 5 = 0x00000005  dos/dos.h: *81
FIBB_READ        #define 3 = 0x00000003  dos/dos.h: *83
FIBB_SCRIPT      #define 6 = 0x00000006  dos/dos.h: *80
FIBB_WRITE       #define 2 = 0x00000002  dos/dos.h: *84
FIBF_ARCHIVE     #define (1<<FIBB_ARCHIVE) = 0x00000010  dos/dos.h: *89
FIBF_DELETE      #define (1<<FIBB_DELETE) = 0x00000001  dos/dos.h: *93
FIBF_EXECUTE     #define (1<<FIBB_EXECUTE) = 0x00000002  dos/dos.h: *92
FIBF_PURE        #define (1<<FIBB_PURE) = 0x00000020  dos/dos.h: *88
FIBF_READ        #define (1<<FIBB_READ) = 0x00000008  dos/dos.h: *90
```

```
FIBF_SCRIPT       #define (1<<FIBB_SCRIPT) = 0x00000040  dos/dos.h: *87
FIBF_WRITE        #define (1<<FIBB_WRITE) = 0x00000004  dos/dos.h: *91
FIL1B_MATCHDIRS   #define 1L = 0x00000001  libraries/asl.h: *132
FIL1B_NOFILES     #define 0L = 0x00000000  libraries/asl.h: *131
FIL1F_MATCHDIRS   #define (1L << FIL1B_MATCHDIRS) = 0x00000002
                  libraries/asl.h: *135
FIL1F_NOFILES     #define (1L << FIL1B_NOFILES) = 0x00000001
                  libraries/asl.h: *134
FILB_DOMSGFUNC    #define 6L = 0x00000006  libraries/asl.h: *110
FILB_DOWILDFUNC   #define 7L = 0x00000007  libraries/asl.h: *108
FILB_MULTISELECT  #define 3L = 0x00000003  libraries/asl.h: *114
FILB_NEWIDCMP     #define 4L = 0x00000004  libraries/asl.h: *113
FILB_PATGAD       #define 0L = 0x00000000  libraries/asl.h: *116
FILB_SAVE         #define 5L = 0x00000005  libraries/asl.h: *112
FILENAME_SIZE     #define 30 = 0x0000001e
                  intuition/preferences.h: *28, 84, 112
FILF_DOMSGFUNC    #define (1L << FILB_DOMSGFUNC) = 0x00000040
                  libraries/asl.h: *120
FILF_DOWILDFUNC   #define (1L << FILB_DOWILDFUNC) = 0x00000080
                  libraries/asl.h: *119
FILF_MULTISELECT  #define (1L << FILB_MULTISELECT) = 0x00000008
                  libraries/asl.h: *124
FILF_NEWIDCMP     #define (1L << FILB_NEWIDCMP) = 0x00000010
                  libraries/asl.h: *123
FILF_PATGAD       #define (1L << FILB_PATGAD) = 0x00000001
                  libraries/asl.h: *125
FILF_SAVE         #define (1L << FILB_SAVE) = 0x00000020  libraries/asl.h: *122
FILLPEN           #define (0x0005) = 0x00000005  intuition/screens.h: *87
FILLRECTCLASS     #define "fillrectclass"  intuition/classusr.h: *46
FILLTEXTPEN       #define (0x0006) = 0x00000006  intuition/screens.h: *88
FILL_CARRYIN      #define 0x4 = 0x00000004  hardware/blit.h: *70
FILL_OR           #define 0x8 = 0x00000008  hardware/blit.h: *68
FILL_XOR          #define 0x10 = 0x00000010  hardware/blit.h: *69
FINE              #define 0x800 = 0x00000800  intuition/preferences.h: *161
FLOAT             typedef float  exec/types.h: *64
FLOYD_DITHERING   #define 0x0400 = 0x00000400  intuition/preferences.h: *257
FOLLOWMOUSE       #define GACT_FOLLOWMOUSE = 0x00000008
                  intuition/iobsolete.h: *72
FONB_BACKCOLOR    #define 1 = 0x00000001  libraries/asl.h: *160
FONB_DOMSGFUNC    #define 6 = 0x00000006  libraries/asl.h: *165
FONB_DOWILDFUNC   #define 7 = 0x00000007  libraries/asl.h: *167
FONB_DRAWMODE     #define 3 = 0x00000003  libraries/asl.h: *162
FONB_FIXEDWIDTH   #define 4 = 0x00000004  libraries/asl.h: *163
FONB_FRONTCOLOR   #define 0 = 0x00000000  libraries/asl.h: *159
FONB_NEWIDCMP     #define 5 = 0x00000005  libraries/asl.h: *164
FONB_STYLES       #define 2 = 0x00000002  libraries/asl.h: *161
FONF_BACKCOLOR    #define (1L << FONB_BACKCOLOR) = 0x00000002
                  libraries/asl.h: *170
FONF_DOMSGFUNC    #define (1L << FONB_DOMSGFUNC) = 0x00000040
                  libraries/asl.h: *175
FONF_DOWILDFUNC   #define (1L << FONB_DOWILDFUNC) = 0x00000080
                  libraries/asl.h: *176
FONF_DRAWMODE     #define (1L << FONB_DRAWMODE) = 0x00000008
                  libraries/asl.h: *172
FONF_FIXEDWIDTH   #define (1L << FONB_FIXEDWIDTH) = 0x00000010
                  libraries/asl.h: *173
FONF_FRONTCOLOR   #define (1L << FONB_FRONTCOLOR) = 0x00000001
                  libraries/asl.h: *169
FONF_NEWIDCMP     #define (1L << FONB_NEWIDCMP) = 0x00000020
                  libraries/asl.h: *174
FONF_STYLES       #define (1L << FONB_STYLES) = 0x00000004
                  libraries/asl.h: *171
FOREVER           #define for(;;)  intuition/intuition.h: *1292
FORMAT_CDN        #define 3 = 0x00000003  dos/datetime.h: *53
FORMAT_DOS        #define 0 = 0x00000000  dos/datetime.h: *50
FORMAT_INT        #define 1 = 0x00000001  dos/datetime.h: *51
```

```
FORMAT_MAX        #define FORMAT_CDN = 0x00000003  dos/datetime.h: *54
FORMAT_USA        #define 2 = 0x00000002  dos/datetime.h: *52
FPB_DESIGNED      #define 6 = 0x00000006  graphics/text.h: *56
FPB_DISKFONT      #define 1 = 0x00000001  graphics/text.h: *46
FPB_PROPORTIONAL  #define 5 = 0x00000005  graphics/text.h: *54
FPB_REMOVED       #define 7 = 0x00000007  graphics/text.h: *63
FPB_REVPATH       #define 2 = 0x00000002  graphics/text.h: *48
FPB_ROMFONT       #define 0 = 0x00000000  graphics/text.h: *44
FPB_TALLDOT       #define 3 = 0x00000003  graphics/text.h: *50
FPB_WIDEDOT       #define 4 = 0x00000004  graphics/text.h: *52
FPF_DESIGNED      #define 0x40 = 0x00000040  graphics/text.h: *61
FPF_DISKFONT      #define 0x02 = 0x00000002  graphics/text.h: *47
FPF_PROPORTIONAL  #define 0x20 = 0x00000020  graphics/text.h: *55
FPF_REMOVED       #define (1<<7) = 0x00000080  graphics/text.h: *64
FPF_REVPATH       #define 0x04 = 0x00000004  graphics/text.h: *49
FPF_ROMFONT       #define 0x01 = 0x00000001  graphics/text.h: *45
FPF_TALLDOT       #define 0x08 = 0x00000008  graphics/text.h: *51
FPF_WIDEDOT       #define 0x10 = 0x00000010  graphics/text.h: *53
FPHALF            #define ((double) 0.5)  libraries/mathffp.h: *28
                  libraries/mathieeedp.h: *29
FPONE             #define ((double) 1.0)  libraries/mathffp.h: *27
                  libraries/mathieeedp.h: *28
FPTEN             #define ((double) 10.0)  libraries/mathffp.h: *26
                  libraries/mathieeedp.h: *27
FPZERO            #define ((double) 0.0)  libraries/mathffp.h: *29
                  libraries/mathieeedp.h: *30
FRAMEF_SPECIFY    #define (1<<0) = 0x00000001  intuition/imageclass.h: *149
FRAMEICLASS       #define "frameiclass"  intuition/classusr.h: *44
FRBUTTONCLASS     #define "frbuttonclass"  intuition/classusr.h: *51
FREEHORIZ         #define 0x0002 = 0x00000002  intuition/intuition.h: *504
FREEVERT          #define 0x0004 = 0x00000004  intuition/intuition.h: *505
FROM_MONITOR      #define 1 = 0x00000001  graphics/monitor.h: *57
FRST_DOT          #define 0x01 = 0x00000001  graphics/rastport.h: *100
FSB_BOLD          #define 1 = 0x00000001  graphics/text.h: *31
FSB_COLORFONT     #define 6 = 0x00000006  graphics/text.h: *38
FSB_EXTENDED      #define 3 = 0x00000003  graphics/text.h: *35
FSB_ITALIC        #define 2 = 0x00000002  graphics/text.h: *33
FSB_TAGGED        #define 7 = 0x00000007  graphics/text.h: *40
FSB_UNDERLINED    #define 0 = 0x00000000  graphics/text.h: *29
FSF_BOLD          #define 0x02 = 0x00000002  graphics/text.h: *32
FSF_COLORFONT     #define 0x40 = 0x00000040  graphics/text.h: *39
FSF_EXTENDED      #define 0x08 = 0x00000008  graphics/text.h: *36
FSF_ITALIC        #define 0x04 = 0x00000004  graphics/text.h: *34
FSF_TAGGED        #define 0x80 = 0x00000080  graphics/text.h: *41
FSF_UNDERLINED    #define 0x01 = 0x00000001  graphics/text.h: *30
FSRNAME           #define "FileSystem.resource"  resources/filesysres.h: *25
FS_NORMAL         #define 0 = 0x00000000  graphics/text.h: *28
FULLMENUNUM       macro (3 arguments)  intuition/intuition.h: *1275
FgPen             char in struct RastPort +0x0019 graphics/rastport.h: *65
FileHandle        structure tag size 0x002c dos/dosextens.h: *91
FileInfoBlock     structure tag size 0x0104 dos/dos.h: *63
                  dos/dosasl.h: 65, 102
FileLock          structure tag size 0x0014 dos/dosextens.h: *443
FileRequester     structure tag size 0x0038 libraries/asl.h: *65
FileSysEntry      structure tag size 0x003e resources/filesysres.h: *33
FileSysHeaderBlock structure tag size 0x0100 devices/hardblocks.h: *156
FileSysResource structure tag size 0x0020 resources/filesysres.h: *27
FileSysStartupMsg structure tag size 0x0010 dos/filehandler.h: *86
FirstCopList      pointer to struct CopList in struct UCopList
           +0x0004 graphics/copper.h: *86
FirstGadget       pointer to struct Gadget in struct Window
           +0x003e intuition/intuition.h: *841
FirstGadget       pointer to struct Gadget in struct NewWindow
           +0x0012 intuition/intuition.h: *990
FirstGadget       pointer to struct Gadget in struct ExtNewWindow
           +0x0012 intuition/intuition.h: *1052
```

Include File Cross Reference Page 53

```
FirstGadget      pointer to struct Gadget in struct Screen
         +0x0146 intuition/screens.h: *135
FirstItem        pointer to struct MenuItem in struct Menu
         +0x0012 intuition/intuition.h: *69
FirstRequest     pointer to struct Requester in struct Window
         +0x0024 intuition/intuition.h: *813
FirstScreen      pointer to struct Screen in struct IntuitionBase
         +0x003c intuition/intuitionbase.h: *80
FirstWindow      pointer to struct Window in struct Screen
         +0x0004 intuition/screens.h: *101
FirstX           short int in struct AreaInfo +0x0014 graphics/rastport.h: *31
FirstY           short int in struct AreaInfo +0x0016 graphics/rastport.h: *31
FlagPtr          pointer to char in struct AreaInfo
         +0x000c graphics/rastport.h: *28
FlagTbl          pointer to char in struct AreaInfo
         +0x0008 graphics/rastport.h: *27
Flags            unsigned char in struct BitMap +0x0004 graphics/gfx.h: *52
Flags            unsigned short int in struct Layer
         +0x001e graphics/clip.h: *42
Flags            unsigned char in struct ColorMap +0x0000 graphics/view.h: *116
Flags            unsigned char in struct GelsInfo
         +0x0001 graphics/rastport.h: *45
Flags            unsigned short int in struct RastPort
         +0x0020 graphics/rastport.h: *72
Flags            unsigned short int in struct Layer_Info
         +0x0058 graphics/layers.h: *43
Flags            unsigned short int in struct Menu
         +0x000c intuition/intuition.h: *67
Flags            unsigned short int in struct MenuItem
         +0x000c intuition/intuition.h: *95
Flags            unsigned short int in struct Requester
         +0x001c intuition/intuition.h: *156
Flags            unsigned short int in struct Gadget
         +0x000c intuition/intuition.h: *223
Flags            unsigned short int in struct BoolInfo
         +0x0000 intuition/intuition.h: *430
Flags            unsigned short int in struct PropInfo
         +0x0000 intuition/intuition.h: *454
Flags            unsigned long int in struct Window
         +0x0018 intuition/intuition.h: *807
Flags            unsigned long int in struct NewWindow
         +0x000e intuition/intuition.h: *983
Flags            unsigned long int in struct ExtNewWindow
         +0x000e intuition/intuition.h: *1051
Flags            unsigned short int in struct Screen
         +0x0014 intuition/screens.h: *108
Flags            short int in struct VSprite +0x0014 graphics/gels.h: *94
Flags            short int in struct Bob +0x0000 graphics/gels.h: *147
Flags            short int in struct AnimComp +0x0000 graphics/gels.h: *175
Flags            unsigned short int in struct GfxBase
         +0x00a8 graphics/gfxbase.h: *46
Flags            unsigned long int in struct IntuitionBase
         +0x0040 intuition/intuitionbase.h: *82
Flags            unsigned char in struct ExpansionBase
         +0x0022 libraries/expansionbase.h: *49
Font             pointer to struct TextFont in struct RastPort
         +0x0034 graphics/rastport.h: *78
Font             pointer to struct TextAttr in struct Screen
         +0x0028 intuition/screens.h: *124
Font             pointer to struct TextAttr in struct NewScreen
         +0x0010 intuition/screens.h: *320
Font             pointer to struct TextAttr in struct ExtNewScreen
         +0x0010 intuition/screens.h: *352
Font             pointer to struct TextFont in struct StringExtend
         +0x0000 intuition/sghooks.h: *21
FontContents     structure tag size 0x0104 libraries/diskfont.h: *30
```

Include File Cross Reference Page 54

```
FontContentsHeader structure tag size 0x0004 libraries/diskfont.h: *54
FontHeight       char in struct Preferences
         +0x0000 intuition/preferences.h: *47
FontRequester    structure tag size 0x0018 libraries/asl.h: *146
FreeClipRects    struct MinList(size 0x000c bytes) in struct Layer_Info
         +0x000c graphics/layers.h: *39
FreeList         structure tag size 0x0010 workbench/workbench.h: *83
FrontPen         unsigned char in struct IntuiText
         +0x0000 intuition/intuition.h: *572
FrontPen         unsigned char in struct Border
         +0x0004 intuition/intuition.h: *601
gelHead          pointer to struct VSprite in struct GelsInfo
         +0x0002 graphics/rastport.h: *46
gelTail          pointer to struct VSprite in struct GelsInfo
         +0x0006 graphics/rastport.h: *46
genloc           array [4] of unsigned short int in struct copinit
         +0x005c graphics/copper.h: *100
gi_Domain        struct IBox(size 0x0008 bytes) in struct GadgetInfo
         +0x0014 intuition/cghooks.h: *49
gi_DrInfo        pointer to struct DrawInfo in struct GadgetInfo
         +0x001e intuition/cghooks.h: *61
gi_Layer         pointer to struct Layer in struct GadgetInfo
         +0x0010 intuition/cghooks.h: *38
gi_Pens          struct (no tag) (size 0x0002 bytes) in struct GadgetInfo
         +0x001c intuition/cghooks.h: *55
gi_RastPort      pointer to struct RastPort in struct GadgetInfo
         +0x000c intuition/cghooks.h: *37
gi_Requester     pointer to struct Requester in struct GadgetInfo
         +0x0008 intuition/cghooks.h: *31
gi_Reserved      array [6] of unsigned long int in struct GadgetInfo
         +0x0022 intuition/cghooks.h: *66
gi_Screen        pointer to struct Screen in struct GadgetInfo
         +0x0000 intuition/cghooks.h: *29
gi_Window        pointer to struct Window in struct GadgetInfo
         +0x0004 intuition/cghooks.h: *30
gpGoInactive     structure tag size 0x000c intuition/gadgetclass.h: *231
gpHitTest        structure tag size 0x000c intuition/gadgetclass.h: *176
gpInput          structure tag size 0x0014 intuition/gadgetclass.h: *201
gpRender         structure tag size 0x0010 intuition/gadgetclass.h: *188
gpgi_Abort       unsigned long int in struct gpGoInactive
         +0x0008 intuition/gadgetclass.h: *236
gpgi_GInfo       pointer to struct GadgetInfo in struct gpGoInactive
         +0x0004 intuition/gadgetclass.h: *233
gpht_GInfo       pointer to struct GadgetInfo in struct gpHitTest
         +0x0004 intuition/gadgetclass.h: *178
gpht_Mouse       struct (no tag) (size 0x0004 bytes) in struct gpHitTest
         +0x0008 intuition/gadgetclass.h: *182
gpi_GInfo        pointer to struct GadgetInfo in struct gpInput
         +0x0004 intuition/gadgetclass.h: *203
gpi_IEvent       pointer to struct InputEvent in struct gpInput
         +0x0008 intuition/gadgetclass.h: *204
gpi_Mouse        struct (no tag) (size 0x0004 bytes) in struct gpInput
         +0x0010 intuition/gadgetclass.h: *209
gpi_Termination pointer to long int in struct gpInput
         +0x000c intuition/gadgetclass.h: *205
gpr_GInfo        pointer to struct GadgetInfo in struct gpRender
         +0x0004 intuition/gadgetclass.h: *190
gpr_RPort        pointer to struct RastPort in struct gpRender
         +0x0008 intuition/gadgetclass.h: *191
gpr_Redraw       long int in struct gpRender
         +0x000c intuition/gadgetclass.h: *192
gpt_Keys         unsigned short int in struct GamePortTrigger
         +0x0000 devices/gameport.h: *39
gpt_Timeout      unsigned short int in struct GamePortTrigger
         +0x0002 devices/gameport.h: *40
gpt_XDelta       unsigned short int in struct GamePortTrigger
```

```
         +0x0004 devices/gameport.h: *41
gpt_YDelta        unsigned short int in struct GamePortTrigger
         +0x0006 devices/gameport.h: *42
gs_Head           struct List(size 0x000e bytes) in struct Layer_Info
         +0x0046 graphics/layers.h: *41
GACT_ACTIVEGADGET #define 0x4000 = 0x00004000  intuition/intuition.h: *385
GACT_ALTKEYMAP    #define 0x1000 = 0x00001000  intuition/intuition.h: *372
GACT_BOOLEXTEND   #define 0x2000 = 0x00002000  intuition/intuition.h: *365
GACT_BORDERSNIFF  #define 0x8000 = 0x00008000  intuition/intuition.h: *362
GACT_BOTTOMBORDER #define 0x0080 = 0x00000080  intuition/intuition.h: *361
GACT_ENDGADGET    #define 0x0004 = 0x00000004  intuition/intuition.h: *342
GACT_FOLLOWMOUSE  #define 0x0008 = 0x00000008  intuition/intuition.h: *352
GACT_IMMEDIATE    #define 0x0002 = 0x00000002  intuition/intuition.h: *336
GACT_LEFTBORDER   #define 0x0020 = 0x00000020  intuition/intuition.h: *359
GACT_LONGINT      #define 0x0800 = 0x00000800  intuition/intuition.h: *371
GACT_RELVERIFY    #define 0x0001 = 0x00000001  intuition/intuition.h: *330
GACT_RIGHTBORDER  #define 0x0010 = 0x00000010  intuition/intuition.h: *358
GACT_STRINGCENTER #define 0x0200 = 0x00000200  intuition/intuition.h: *369
GACT_STRINGEXTEND #define 0x2000 = 0x00002000  intuition/intuition.h: *373
GACT_STRINGLEFT   #define 0x0000 = 0x00000000  intuition/intuition.h: *368
GACT_STRINGRIGHT  #define 0x0400 = 0x00000400  intuition/intuition.h: *370
GACT_TOGGLESELECT #define 0x0100 = 0x00000100  intuition/intuition.h: *364
GACT_TOPBORDER    #define 0x0040 = 0x00000040  intuition/intuition.h: *360
GADGBACKFILL      #define 0x0001 = 0x00000001  workbench/workbench.h: *111
GADGDISABLED      #define GFLG_DISABLED = 0x00000100
                  intuition/iobsolete.h: *60
GADGET0002        #define GTYP_GADGET0002 = 0x00000002
                  intuition/iobsolete.h: *105
GADGETCLASS       #define "gadgetclass"  intuition/classusr.h: *47
GADGETCOUNT       #define 8 = 0x00000008  intuition/intuitionbase.h: *46
GADGETDOWN        #define IDCMP_GADGETDOWN = 0x00000020
                  intuition/iobsolete.h: *119
GADGETTYPE        #define GTYP_GADGETTYPE = 0x0000fc00
                  intuition/iobsolete.h: *91
GADGETUP          #define IDCMP_GADGETUP = 0x00000040
                  intuition/iobsolete.h: *120
GADGET_BOX        macro (1 argument)  intuition/imageclass.h: *28
GADGHBOX          #define GFLG_GADGHBOX = 0x00000001
                  intuition/iobsolete.h: *51
GADGHCOMP         #define GFLG_GADGHCOMP = 0x00000000
                  intuition/iobsolete.h: *50
GADGHIGHBITS      #define GFLG_GADGHIGHBITS = 0x00000003
                  intuition/iobsolete.h: *49
GADGHIMAGE        #define GFLG_GADGHIMAGE = 0x00000002
                  intuition/iobsolete.h: *52
GADGHNONE         #define GFLG_GADGHNONE = 0x00000003
                  intuition/iobsolete.h: *53
GADGIMAGE         #define GFLG_GADGIMAGE = 0x00000004
                  intuition/iobsolete.h: *54
GADGIMMEDIATE     #define GACT_IMMEDIATE = 0x00000002
                  intuition/iobsolete.h: *70
GADTOOLBIT        #define (0x8000) = 0x00008000  libraries/gadtools.h: *55
GADTOOLMASK       #define (~GADTOOLBIT) = 0xffff7fff  libraries/gadtools.h: *57
GA_BORDER         #define GA_Border = 0x8003000b  intuition/iobsolete.h: *190
GA_BOTTOMBORDER   #define GA_BottomBorder = 0x8003001b
                  intuition/iobsolete.h: *205
GA_Border         #define (GA_Dummy + 0x000B) = 0x8003000b
                  intuition/gadgetclass.h: *44
GA_BottomBorder   #define (GA_Dummy + 0x001B) = 0x8003001b
                  intuition/gadgetclass.h: *60
GA_DISABLED       #define GA_Disabled = 0x8003000e  intuition/iobsolete.h: *193
GA_DRAWINFO       #define GA_DrawInfo = 0x80030021  intuition/iobsolete.h: *211
GA_Disabled       #define (GA_Dummy + 0x000E) = 0x8003000e
                  intuition/gadgetclass.h: *47
GA_DrawInfo       #define (GA_Dummy + 0x0021) = 0x80030021
                  intuition/gadgetclass.h: *79
```

```
GA_Dummy          #define (TAG_USER +0x30000) = 0x80030000
                  intuition/gadgetclass.h: *33
GA_ENDGADGET      #define GA_EndGadget = 0x80030014
                  intuition/iobsolete.h: *198
GA_EndGadget      #define (GA_Dummy + 0x0014) = 0x80030014
                  intuition/gadgetclass.h: *53
GA_FOLLOWMOUSE    #define GA_FollowMouse = 0x80030017
                  intuition/iobsolete.h: *201
GA_FollowMouse    #define (GA_Dummy + 0x0017) = 0x80030017
                  intuition/gadgetclass.h: *56
GA_GZZGADGET      #define GA_GZZGadget = 0x8003000f
                  intuition/iobsolete.h: *194
GA_GZZGadget      #define (GA_Dummy + 0x000F) = 0x8003000f
                  intuition/gadgetclass.h: *48
GA_HEIGHT         #define GA_Height = 0x80030007  intuition/iobsolete.h: *186
GA_HIGHLIGHT      #define GA_Highlight = 0x8003000d
                  intuition/iobsolete.h: *192
GA_Height         #define (GA_Dummy + 0x0007) = 0x80030007
                  intuition/gadgetclass.h: *40
GA_Highlight      #define (GA_Dummy + 0x000D) = 0x8003000d
                  intuition/gadgetclass.h: *46
GA_ID             #define (GA_Dummy + 0x0010) = 0x80030010
                  intuition/gadgetclass.h: *49
GA_IMAGE          #define GA_Image = 0x8003000a  intuition/iobsolete.h: *189
GA_IMMEDIATE      #define GA_Immediate = 0x80030015
                  intuition/iobsolete.h: *199
GA_INTUITEXT      #define GA_IntuiText = 0x80030022
                  intuition/iobsolete.h: *212
GA_Image          #define (GA_Dummy + 0x000A) = 0x8003000a
                  intuition/gadgetclass.h: *43
GA_Immediate      #define (GA_Dummy + 0x0015) = 0x80030015
                  intuition/gadgetclass.h: *54
GA_IntuiText      #define (GA_Dummy + 0x0022) = 0x80030022
                  intuition/gadgetclass.h: *85
GA_LABELIMAGE     #define GA_LabelImage = 0x80030023
                  intuition/iobsolete.h: *213
GA_LEFT           #define GA_Left = 0x80030001  intuition/iobsolete.h: *180
GA_LEFTBORDER     #define GA_LeftBorder = 0x80030019
                  intuition/iobsolete.h: *203
GA_LabelImage     #define (GA_Dummy + 0x0023) = 0x80030023
                  intuition/gadgetclass.h: *88
GA_Left           #define (GA_Dummy + 0x0001) = 0x80030001
                  intuition/gadgetclass.h: *34
GA_LeftBorder     #define (GA_Dummy + 0x0019) = 0x80030019
                  intuition/gadgetclass.h: *58
GA_NEXT           #define GA_Next = 0x80030020  intuition/iobsolete.h: *210
GA_Next           #define (GA_Dummy + 0x0020) = 0x80030020
                  intuition/gadgetclass.h: *76
GA_PREVIOUS       #define GA_Previous = 0x8003001f  intuition/iobsolete.h: *209
GA_Previous       #define (GA_Dummy + 0x001F) = 0x8003001f
                  intuition/gadgetclass.h: *69
GA_RELBOTTOM      #define GA_RelBottom = 0x80030004
                  intuition/iobsolete.h: *183
GA_RELHEIGHT      #define GA_RelHeight = 0x80030008
                  intuition/iobsolete.h: *187
GA_RELRIGHT       #define GA_RelRight = 0x80030002  intuition/iobsolete.h: *181
GA_RELVERIFY      #define GA_RelVerify = 0x80030016
                  intuition/iobsolete.h: *200
GA_RELWIDTH       #define GA_RelWidth = 0x80030006  intuition/iobsolete.h: *185
GA_RIGHTBORDER    #define GA_RightBorder = 0x80030018
                  intuition/iobsolete.h: *202
GA_RelBottom      #define (GA_Dummy + 0x0004) = 0x80030004
                  intuition/gadgetclass.h: *37
GA_RelHeight      #define (GA_Dummy + 0x0008) = 0x80030008
                  intuition/gadgetclass.h: *41
GA_RelRight       #define (GA_Dummy + 0x0002) = 0x80030002
```

```
                     intuition/gadgetclass.h: *35
GA_RelVerify         #define (GA_Dummy + 0x0016) = 0x80030016
                     intuition/gadgetclass.h: *55
GA_RelWidth          #define (GA_Dummy + 0x0006) = 0x80030006
                     intuition/gadgetclass.h: *39
GA_RightBorder       #define (GA_Dummy + 0x0018) = 0x80030018
                     intuition/gadgetclass.h: *57
GA_SELECTED          #define GA_Selected = 0x80030013  intuition/iobsolete.h: *197
GA_SELECTRENDER      #define GA_SelectRender = 0x8003000c
                     intuition/iobsolete.h: *191
GA_SPECIALINFO       #define GA_SpecialInfo = 0x80030012
                     intuition/iobsolete.h: *196
GA_SYSGADGET         #define GA_SysGadget = 0x8003001d
                     intuition/iobsolete.h: *207
GA_SYSGTYPE          #define GA_SysGType = 0x8003001e  intuition/iobsolete.h: *208
GA_SelectRender      #define (GA_Dummy + 0x000C) = 0x8003000c
                     intuition/gadgetclass.h: *45
GA_Selected          #define (GA_Dummy + 0x0013) = 0x80030013
                     intuition/gadgetclass.h: *52
GA_SpecialInfo       #define (GA_Dummy + 0x0012) = 0x80030012
                     intuition/gadgetclass.h: *51
GA_SysGType          #define (GA_Dummy + 0x001E) = 0x8003001e
                     intuition/gadgetclass.h: *66
GA_SysGadget         #define (GA_Dummy + 0x001D) = 0x8003001d
                     intuition/gadgetclass.h: *64
GA_TEXT              #define GA_Text = 0x80030009  intuition/iobsolete.h: *188
GA_TOGGLESELECT      #define GA_ToggleSelect = 0x8003001c
                     intuition/iobsolete.h: *206
GA_TOP               #define GA_Top = 0x80030003  intuition/iobsolete.h: *182
GA_TOPBORDER         #define GA_TopBorder = 0x8003001a
                     intuition/iobsolete.h: *204
GA_TabCycle          #define (GA_Dummy + 0x0024) = 0x80030024
                     intuition/gadgetclass.h: *93
GA_Text              #define (GA_Dummy + 0x0009) = 0x80030009
                     intuition/gadgetclass.h: *42
GA_ToggleSelect      #define (GA_Dummy + 0x001C) = 0x8003001c
                     intuition/gadgetclass.h: *61
GA_Top               #define (GA_Dummy + 0x0003) = 0x80030003
                     intuition/gadgetclass.h: *36
GA_TopBorder         #define (GA_Dummy + 0x001A) = 0x8003001a
                     intuition/gadgetclass.h: *59
GA_USERDATA          #define GA_UserData = 0x80030011  intuition/iobsolete.h: *195
GA_UserData          #define (GA_Dummy + 0x0011) = 0x80030011
                     intuition/gadgetclass.h: *50
GA_WIDTH             #define GA_Width = 0x80030005  intuition/iobsolete.h: *184
GA_Width             #define (GA_Dummy + 0x0005) = 0x80030005
                     intuition/gadgetclass.h: *38
GELGONE              #define 0x0400 = 0x00000400  graphics/gels.h: *29
GENERIC_KIND         #define 0 = 0x00000000  libraries/gadtools.h: *34
GENLOC               #define 2 = 0x00000002  graphics/gfxbase.h: *96
GENLOCK_AUDIO        #define 0x0100 = 0x00000100  graphics/view.h: *95
GENLOCK_VIDEO        #define 0x0002 = 0x00000002  graphics/view.h: *90
GFLG_DISABLED        #define 0x0100 = 0x00000100  intuition/intuition.h: *297
GFLG_GADGHBOX        #define 0x0001 = 0x00000001  intuition/intuition.h: *269
GFLG_GADGHCOMP       #define 0x0000 = 0x00000000  intuition/intuition.h: *268
GFLG_GADGHIGHBITS #define 0x0003 = 0x00000003  intuition/intuition.h: *267
GFLG_GADGHIMAGE      #define 0x0002 = 0x00000002  intuition/intuition.h: *270
GFLG_GADGHNONE       #define 0x0003 = 0x00000003  intuition/intuition.h: *271
GFLG_GADGIMAGE       #define 0x0004 = 0x00000004  intuition/intuition.h: *276
GFLG_LABELIMAGE      #define 0x2000 = 0x00002000  intuition/intuition.h: *310
GFLG_LABELITEXT      #define 0x0000 = 0x00000000  intuition/intuition.h: *308
GFLG_LABELMASK       #define 0x3000 = 0x00003000  intuition/intuition.h: *307
GFLG_LABELSTRING     #define 0x1000 = 0x00001000  intuition/intuition.h: *309
GFLG_RELBOTTOM       #define 0x0008 = 0x00000008  intuition/intuition.h: *286
GFLG_RELHEIGHT       #define 0x0040 = 0x00000040  intuition/intuition.h: *289
GFLG_RELRIGHT        #define 0x0010 = 0x00000010  intuition/intuition.h: *287
```

```
GFLG_RELWIDTH        #define 0x0020 = 0x00000020  intuition/intuition.h: *288
GFLG_SELECTED        #define 0x0080 = 0x00000080  intuition/intuition.h: *291
GFLG_STRINGEXTEND #define 0x0400 = 0x00000400  intuition/intuition.h: *323
GFLG_TABCYCLE        #define 0x0200 = 0x00000200  intuition/intuition.h: *315
GFXB_BIG_BLITS       #define 0 = 0x00000000  graphics/gfxbase.h: *103
GFXB_HR_AGNUS        #define 0 = 0x00000000  graphics/gfxbase.h: *104
GFXB_HR_DENISE       #define 1 = 0x00000001  graphics/gfxbase.h: *105
GFXF_BIG_BLITS       #define 1 = 0x00000001  graphics/gfxbase.h: *107
GFXF_HR_AGNUS        #define 1 = 0x00000001  graphics/gfxbase.h: *108
GFXF_HR_DENISE       #define 2 = 0x00000002  graphics/gfxbase.h: *109
GIMMEZEROZERO        #define WFLG_GIMMEZEROZERO = 0x00000400
                     intuition/iobsolete.h: *158
GLOBAL               #define extern  exec/types.h: *20
GLOBALSZ             #define 200 = 0x000000c8  rexx/storage.h: *185, 188
GMR_GADGETHIT        #define (0x00000004) = 0x00000004
                     intuition/gadgetclass.h: *185
GMR_MEACTIVE         #define (0) = 0x00000000  intuition/gadgetclass.h: *217
GMR_NEXTACTIVE       #define (1 << 4) = 0x00000010  intuition/gadgetclass.h: *227
GMR_NOREUSE          #define (1 << 1) = 0x00000002  intuition/gadgetclass.h: *218
GMR_PREVACTIVE       #define (1 << 5) = 0x00000020  intuition/gadgetclass.h: *228
GMR_REUSE            #define (1 << 2) = 0x00000004  intuition/gadgetclass.h: *219
GMR_VERIFY           #define (1 << 3) = 0x00000008  intuition/gadgetclass.h: *220
GM_Dummy             #define (-1) = 0xffffffff  intuition/gadgetclass.h: *164
GM_GOACTIVE          #define (2) = 0x00000002  intuition/gadgetclass.h: *169
GM_GOINACTIVE        #define (4) = 0x00000004  intuition/gadgetclass.h: *171
GM_HANDLEINPUT       #define (3) = 0x00000003  intuition/gadgetclass.h: *170
GM_HITTEST           #define (0) = 0x00000000  intuition/gadgetclass.h: *167
GM_RENDER            #define (1) = 0x00000001  intuition/gadgetclass.h: *168
GPCT_ABSJOYSTICK #define 3 = 0x00000003  devices/gameport.h: *51
GPCT_ALLOCATED       #define -1 = 0xffffffff  devices/gameport.h: *46
GPCT_MOUSE           #define 1 = 0x00000001  devices/gameport.h: *49
GPCT_NOCONTROLLER #define 0 = 0x00000000  devices/gameport.h: *47
GPCT_RELJOYSTICK #define 2 = 0x00000002  devices/gameport.h: *50
GPDERR_SETCTYPE      #define 1 = 0x00000001  devices/gameport.h: *55
GPD_ASKCTYPE         #define (CMD_NONSTD+1) = 0x0000000a  devices/gameport.h: *25
GPD_ASKTRIGGER       #define (CMD_NONSTD+3) = 0x0000000c  devices/gameport.h: *27
GPD_READEVENT        #define (CMD_NONSTD+0) = 0x00000009  devices/gameport.h: *24
GPD_SETCTYPE         #define (CMD_NONSTD+2) = 0x0000000b  devices/gameport.h: *26
GPD_SETTRIGGER       #define (CMD_NONSTD+4) = 0x0000000d  devices/gameport.h: *28
GPTB_DOWNKEYS        #define 0 = 0x00000000  devices/gameport.h: *33
GPTB_UPKEYS          #define 1 = 0x00000001  devices/gameport.h: *35
GPTF_DOWNKEYS        #define (1<<0) = 0x00000001  devices/gameport.h: *34
GPTF_UPKEYS          #define (1<<1) = 0x00000002  devices/gameport.h: *36
GRAPHICSNAME         #define "graphics.library"  graphics/gfxbase.h: *111
GRAPHICS_CLIP_H      #define  graphics/clip.h: *2, 1
                     intuition/screens.h: 23
GRAPHICS_COLLIDE_H #define  graphics/collide.h: *2
GRAPHICS_COPPER_H #define  graphics/copper.h: *2, 1
GRAPHICS_DISPLAYINFO_H #define  graphics/displayinfo.h: *2
GRAPHICS_DISPLAY_H #define  graphics/display.h: *2
GRAPHICS_GELS_H   #define  graphics/gels.h: *2
GRAPHICS_GFXBASE_H #define  graphics/gfxbase.h: *2
GRAPHICS_GFXMACROS_H #define  graphics/gfxmacros.h: *2
GRAPHICS_GFXNODES_H #define  graphics/gfxnodes.h: *2, 1
                     graphics/monitor.h: 19
GRAPHICS_GFX_H       #define  graphics/gfx.h: *2, 1
                     graphics/clip.h: 19
                     graphics/view.h: 21
                     graphics/monitor.h: 23
                     graphics/rastport.h: 19
                     graphics/text.h: 19
                     intuition/screens.h: 19
                     graphics/displayinfo.h: 19
                     graphics/regions.h: 19
GRAPHICS_GRAPHINT_H #define  graphics/graphint.h: *2
GRAPHICS_LAYERS_H #define  graphics/layers.h: *2, 1
```

```
                     intuition/screens.h: 35
GRAPHICS_MONITOR_H #define  graphics/monitor.h: *2, 1
                     graphics/displayinfo.h: 23
GRAPHICS_RASTPORT_H #define  graphics/rastport.h: *2, 1
                     intuition/screens.h: 31
                     graphics/gfxmacros.h: 19
GRAPHICS_REGIONS_H #define  graphics/regions.h: *2
GRAPHICS_SCALE_H #define  graphics/scale.h: *2
GRAPHICS_SPRITE_H #define  graphics/sprite.h: *2
GRAPHICS_TEXT_H  #define  graphics/text.h: *2, 1
                     libraries/asl.h: 40
                     libraries/diskfont.h: 24
GRAPHICS_VIDEOCONTROL_H #define  graphics/videocontrol.h: *2
GRAPHICS_VIEW_H   #define  graphics/view.h: *2, 1
                     intuition/screens.h: 27
GREDRAW_REDRAW    #define (1) = 0x00000001  intuition/gadgetclass.h: *197
GREDRAW_TOGGLE    #define (0) = 0x00000000  intuition/gadgetclass.h: *198
GREDRAW_UPDATE    #define (2) = 0x00000002  intuition/gadgetclass.h: *196
GRELBOTTOM        #define GFLG_RELBOTTOM = 0x00000008
                     intuition/iobsolete.h: *55
GRELHEIGHT        #define GFLG_RELHEIGHT = 0x00000040
                     intuition/iobsolete.h: *58
GRELRIGHT         #define GFLG_RELRIGHT = 0x00000010
                     intuition/iobsolete.h: *56
GRELWIDTH         #define GFLG_RELWIDTH = 0x00000020
                     intuition/iobsolete.h: *57
GREY_SCALE2       #define 0x1000 = 0x00001000  intuition/preferences.h: *260
GROUPGCLASS       #define "groupgclass"  intuition/classusr.h: *52
GTBB_Recessed     #define GT_TagBase+51 = 0x80080033
                     libraries/gadtools.h: *246
GTCB_Checked      #define GT_TagBase+4 = 0x80080004  libraries/gadtools.h: *195
GTCY_Active       #define GT_TagBase+15 = 0x8008000f
                     libraries/gadtools.h: *213
GTCY_Labels       #define GT_TagBase+14 = 0x8008000e
                     libraries/gadtools.h: *212
GTIN_MaxChars     #define GT_TagBase+48 = 0x80080030
                     libraries/gadtools.h: *241
GTIN_Number       #define GT_TagBase+47 = 0x8008002f
                     libraries/gadtools.h: *240
GTLV_Labels       #define GT_TagBase+6 = 0x80080006  libraries/gadtools.h: *198
GTLV_ReadOnly     #define GT_TagBase+7 = 0x80080007  libraries/gadtools.h: *200
GTLV_ScrollWidth  #define GT_TagBase+8 = 0x80080008  libraries/gadtools.h: *201
GTLV_Selected     #define GT_TagBase+54 = 0x80080036
                     libraries/gadtools.h: *254
GTLV_ShowSelected #define GT_TagBase+53 = 0x80080035
                     libraries/gadtools.h: *252
GTLV_Top          #define GT_TagBase+5 = 0x80080005  libraries/gadtools.h: *197
GTMENUITEM_USERDATA macro (1 argument)   libraries/gadtools.h: *173
GTMENU_INVALID    #define 0x00000002 = 0x00000002  libraries/gadtools.h: *181
GTMENU_NOMEM      #define 0x00000003 = 0x00000003  libraries/gadtools.h: *182
GTMENU_TRIMMED    #define 0x00000001 = 0x00000001  libraries/gadtools.h: *180
GTMENU_USERDATA macro (1 argument)   libraries/gadtools.h: *172
GTMN_FrontPen     #define GT_TagBase+50 = 0x80080032
                     libraries/gadtools.h: *244
GTMN_FullMenu     #define GT_TagBase+62 = 0x8008003e
                     libraries/gadtools.h: *274
GTMN_Menu         #define GT_TagBase+60 = 0x8008003c
                     libraries/gadtools.h: *267
GTMN_SecondaryError #define GT_TagBase+63 = 0x8008003f
                     libraries/gadtools.h: *276
GTMN_TextAttr     #define GT_TagBase+49 = 0x80080031
                     libraries/gadtools.h: *243
GTMX_Active       #define GT_TagBase+10 = 0x8008000a
                     libraries/gadtools.h: *204
GTMX_Labels       #define GT_TagBase+9 = 0x80080009  libraries/gadtools.h: *203
GTMX_Spacing      #define GT_TagBase+61 = 0x8008003d
```

```
                     libraries/gadtools.h: *270
GTNM_Border       #define GT_TagBase+58 = 0x8008003a
                     libraries/gadtools.h: *261
GTNM_Number       #define GT_TagBase+13 = 0x8008000d
                     libraries/gadtools.h: *210
GTNW_Active       #define GTCY_Active = 0x8008000f  libraries/gadtools.h: *289
GTNW_Labels       #define GTCY_Labels = 0x8008000e  libraries/gadtools.h: *288
GTPA_Color        #define GT_TagBase+17 = 0x80080011
                     libraries/gadtools.h: *216
GTPA_ColorOffset  #define GT_TagBase+18 = 0x80080012
                     libraries/gadtools.h: *217
GTPA_Depth        #define GT_TagBase+16 = 0x80080010
                     libraries/gadtools.h: *215
GTPA_IndicatorHeight #define GT_TagBase+20 = 0x80080014
                     libraries/gadtools.h: *219
GTPA_IndicatorWidth #define GT_TagBase+19 = 0x80080013
                     libraries/gadtools.h: *218
GTSC_Arrows       #define GT_TagBase+59 = 0x8008003b
                     libraries/gadtools.h: *264
GTSC_Overlap      #define GT_TagBase+24 = 0x80080018
                     libraries/gadtools.h: *224
GTSC_Top          #define GT_TagBase+21 = 0x80080015
                     libraries/gadtools.h: *221
GTSC_Total        #define GT_TagBase+22 = 0x80080016
                     libraries/gadtools.h: *222
GTSC_Visible      #define GT_TagBase+23 = 0x80080017
                     libraries/gadtools.h: *223
GTSL_DispFunc     #define GT_TagBase+44 = 0x8008002c
                     libraries/gadtools.h: *235
GTSL_Level        #define GT_TagBase+40 = 0x80080028
                     libraries/gadtools.h: *230
GTSL_LevelFormat  #define GT_TagBase+42 = 0x8008002a
                     libraries/gadtools.h: *232
GTSL_LevelPlace   #define GT_TagBase+43 = 0x8008002b
                     libraries/gadtools.h: *233
GTSL_Max          #define GT_TagBase+39 = 0x80080027
                     libraries/gadtools.h: *229
GTSL_MaxLevelLen  #define GT_TagBase+41 = 0x80080029
                     libraries/gadtools.h: *231
GTSL_Min          #define GT_TagBase+38 = 0x80080026
                     libraries/gadtools.h: *228
GTST_MaxChars     #define GT_TagBase+46 = 0x8008002e
                     libraries/gadtools.h: *238
GTST_String       #define GT_TagBase+45 = 0x8008002d
                     libraries/gadtools.h: *237
GTTX_Border       #define GT_TagBase+57 = 0x80080039
                     libraries/gadtools.h: *259
GTTX_CopyText     #define GT_TagBase+12 = 0x8008000c
                     libraries/gadtools.h: *208
GTTX_Text         #define GT_TagBase+11 = 0x8008000b
                     libraries/gadtools.h: *206
GTVI_NWTags       #define GT_TagBase+2 = 0x80080002  libraries/gadtools.h: *191
GTVI_NewWindow    #define GT_TagBase+1 = 0x80080001  libraries/gadtools.h: *190
GTYPEMASK         #define GTYP_GTYPEMASK = 0x00000007
                     intuition/iobsolete.h: *109
GTYP_BOOLGADGET   #define 0x0001 = 0x00000001  intuition/intuition.h: *410
GTYP_CLOSE        #define 0x0080 = 0x00000080  intuition/intuition.h: *408
GTYP_CUSTOMGADGET #define 0x0005 = 0x00000005  intuition/intuition.h: *414
GTYP_GADGET0002   #define 0x0002 = 0x00000002  intuition/intuition.h: *411
GTYP_GADGETTYPE   #define 0xFC00 = 0x0000fc00  intuition/intuition.h: *395
GTYP_GTYPEMASK    #define 0x0007 = 0x00000007  intuition/intuition.h: *415
GTYP_GZZGADGET    #define 0x2000 = 0x00002000  intuition/intuition.h: *398
GTYP_PROPGADGET   #define 0x0003 = 0x00000003  intuition/intuition.h: *412
GTYP_REQGADGET    #define 0x1000 = 0x00001000  intuition/intuition.h: *399
GTYP_SCRGADGET    #define 0x4000 = 0x00004000  intuition/intuition.h: *397
GTYP_SDOWNBACK    #define 0x0070 = 0x00000070  intuition/intuition.h: *407
```

Include File Cross Reference Page 61

```
GTYP_SDRAGGING   #define 0x0030 = 0x00000030  intuition/intuition.h: *403
GTYP_SIZING      #define 0x0010 = 0x00000010  intuition/intuition.h: *401
GTYP_STRGADGET   #define 0x0004 = 0x00000004  intuition/intuition.h: *413
GTYP_SUPFRONT    #define 0x0050 = 0x00000050  intuition/intuition.h: *405
GTYP_SYSGADGET   #define 0x8000 = 0x00008000  intuition/intuition.h: *396
GTYP_WDOWNBACK   #define 0x0060 = 0x00000060  intuition/intuition.h: *406
GTYP_WDRAGGING   #define 0x0020 = 0x00000020  intuition/intuition.h: *402
GTYP_WUPFRONT    #define 0x0040 = 0x00000040  intuition/intuition.h: *404
GT_Private0      #define GT_TagBase+3 = 0x80080003  libraries/gadtools.h: *193
GT_Reserved0     #define GT_TagBase+55 = 0x80080037
                 libraries/gadtools.h: *255
GT_Reserved1     #define GT_TagBase+56 = 0x80080038
                 libraries/gadtools.h: *256
GT_TagBase       #define TAG_USER + 0x80000 = 0x80080000
                 libraries/gadtools.h: *188
GT_Underscore    #define GT_TagBase+64 = 0x80080040
                 libraries/gadtools.h: *279
GT_VisualInfo    #define GT_TagBase+52 = 0x80080034
                 libraries/gadtools.h: *248
GVB_BINARY_VAR   #define 10 = 0x0000000a  dos/var.h: *53
GVB_GLOBAL_ONLY  #define 8 = 0x00000008  dos/var.h: *49
GVB_LOCAL_ONLY   #define 9 = 0x00000009  dos/var.h: *51
GVF_BINARY_VAR   #define 0x400 = 0x00000400  dos/var.h: *54
GVF_GLOBAL_ONLY  #define 0x100 = 0x00000100  dos/var.h: *50
GVF_LOCAL_ONLY   #define 0x200 = 0x00000200  dos/var.h: *52
GZZGADGET        #define GTYP_GZZGADGET = 0x00002000
                 intuition/iobsolete.h: *94
GZZHeight        short int in struct Window +0x0072 intuition/intuition.h: *881
GZZMouseX        short int in struct Window +0x006c intuition/intuition.h: *875
GZZMouseY        short int in struct Window +0x006e intuition/intuition.h: *876
GZZWidth         short int in struct Window +0x0070 intuition/intuition.h: *880
Gadget           structure tag
     size 0x002c intuition/intuition.h: *153, 216, 218, 841, 990,
                             1052
                 intuition/screens.h: 135, 324, 354
                 intuition/sghooks.h: 35
                 workbench/workbench.h: 65
Gadget           pointer to struct Gadget in struct SGWork
         +0x0000 intuition/sghooks.h: *35
GadgetID         unsigned short int in struct Gadget
         +0x0026 intuition/intuition.h: *260
GadgetInfo       structure tag size 0x003a intuition/cghooks.h: *27
                 intuition/classusr.h: 81, 91
                 intuition/gadgetclass.h: 178, 190, 203, 233
                 intuition/sghooks.h: 49
GadgetInfo       pointer to struct GadgetInfo in struct SGWork
         +0x0026 intuition/sghooks.h: *49
GadgetRender     pointer to void in struct Gadget
         +0x0012 intuition/intuition.h: *233
GadgetText       pointer to struct IntuiText in struct Gadget
         +0x001a intuition/intuition.h: *240
GadgetType       unsigned short int in struct Gadget
         +0x0010 intuition/intuition.h: *227
Gadgets          pointer to struct Gadget in struct NewScreen
         +0x0018 intuition/screens.h: *324
Gadgets          pointer to struct Gadget in struct ExtNewScreen
         +0x0018 intuition/screens.h: *354
GamePortTrigger structure tag size 0x0008 devices/gameport.h: *38
GelsInfo         structure tag size 0x0026 graphics/rastport.h: *41, 63
GelsInfo         pointer to struct GelsInfo in struct RastPort
         +0x0014 graphics/rastport.h: *63
GfxBase          structure tag size 0x01a6 graphics/gfxbase.h: *25
Green            unsigned short int in struct ColorSpec
         +0x0004 intuition/intuition.h: *1244
h_Data           pointer to void in struct Hook +0x0010 utility/hooks.h: *28
h_Entry          pointer to function returning unsigned long int in struct
```

Include File Cross Reference Page 62

```
                 Hook
         +0x0008 utility/hooks.h: *26
h_MinNode        struct MinNode(size 0x0008 bytes) in struct Hook
         +0x0000 utility/hooks.h: *25
h_SubEntry       pointer to function returning unsigned long int in struct
                 Hook
         +0x000c utility/hooks.h: *27
hash_table       pointer to long int in struct GfxBase
         +0x015a graphics/gfxbase.h: *76
hblank           struct AnalogSignalInterval(size 0x0004 bytes) in struct
                 SpecialMonitor
         +0x002a graphics/monitor.h: *150
hbstop           unsigned short int in struct Custom
         +0x01c6 hardware/custom.h: *126
hbstrt           unsigned short int in struct Custom
         +0x01c4 hardware/custom.h: *125
hcenter          unsigned short int in struct Custom
         +0x01e2 hardware/custom.h: *140
hedley           array [8] of unsigned long int in struct GfxBase
         +0x00f4 graphics/gfxbase.h: *70
hedley_count     short int in struct GfxBase +0x0154 graphics/gfxbase.h: *73
hedley_flags     unsigned short int in struct GfxBase
         +0x0156 graphics/gfxbase.h: *74
hedley_hint      unsigned char in struct GfxBase
         +0x0162 graphics/gfxbase.h: *79
hedley_hint2     unsigned char in struct GfxBase
         +0x0163 graphics/gfxbase.h: *80
hedley_sprites   array [8] of unsigned long int in struct GfxBase
         +0x0114 graphics/gfxbase.h: *71
hedley_sprites1 array [8] of unsigned long int in struct GfxBase
         +0x0134 graphics/gfxbase.h: *72
hedley_tmp       short int in struct GfxBase +0x0158 graphics/gfxbase.h: *75
height           unsigned char in struct mouth_rb
         +0x0059 devices/narrator.h: *131
height           unsigned short int in struct SimpleSprite
         +0x0004 graphics/sprite.h: *24
hhposr           unsigned short int in struct Custom
         +0x01da hardware/custom.h: *136
hhposw           unsigned short int in struct Custom
         +0x01d8 hardware/custom.h: *135
hifillPen        #define FILLPEN = 0x00000005  intuition/iobsolete.h: *266
hifilltextPen    #define FILLTEXTPEN = 0x00000006  intuition/iobsolete.h: *267
hilighttextPen   #define HIGHLIGHTTEXTPEN = 0x00000008
                 intuition/iobsolete.h: *269
hour             unsigned short int in struct ClockData
         +0x0004 utility/date.h: *22
hsstop           unsigned short int in struct Custom
         +0x01c2 hardware/custom.h: *124
hsstrt           unsigned short int in struct Custom
         +0x01de hardware/custom.h: *138
hsync            struct AnalogSignalInterval(size 0x0004 bytes) in struct
                 SpecialMonitor
         +0x0032 graphics/monitor.h: *152
htotal           unsigned short int in struct Custom
         +0x01c0 hardware/custom.h: *123
HALFTONE_DITHERING #define 0x0200 = 0x00000200  intuition/preferences.h: *256
HAM              #define 0x0800 = 0x00000800  graphics/view.h: *97
HAMLACE_KEY      #define 0x00000804 = 0x00000804  graphics/displayinfo.h: *162
HAM_KEY          #define 0x00000800 = 0x00000800  graphics/displayinfo.h: *158
HARDWARE_ADKBITS_H #define  hardware/adkbits.h: *2
HARDWARE_BLIT_H  #define  hardware/blit.h: *2
HARDWARE_CIA_H   #define  hardware/cia.h: *2
HARDWARE_CUSTOM_H #define  hardware/custom.h: *2, 1
HARDWARE_DMABITS_H #define  hardware/dmabits.h: *2
HARDWARE_INTBITS_H #define  hardware/intbits.h: *2
HASZOOM          #define WFLG_HASZOOM = 0x20000000
```

```
                   intuition/iobsolete.h: *172
HD_SCSICMD         #define 28 = 0x0000001c  devices/scsidisk.h: *68
HFERR_BadStatus    #define 45 = 0x0000002d  devices/scsidisk.h: *114
HFERR_DMA          #define 41 = 0x00000029  devices/scsidisk.h: *110
HFERR_NoBoard      #define 50 = 0x00000032  devices/scsidisk.h: *117
HFERR_Parity       #define 43 = 0x0000002b  devices/scsidisk.h: *112
HFERR_Phase        #define 42 = 0x0000002a  devices/scsidisk.h: *111
HFERR_SelTimeout   #define 44 = 0x0000002c  devices/scsidisk.h: *113
HFERR_SelfUnit     #define 40 = 0x00000028  devices/scsidisk.h: *109
HIGHBOX            #define 0x0080 = 0x00000080  intuition/intuition.h: *128
HIGHCOMP           #define 0x0040 = 0x00000040  intuition/intuition.h: *127
HIGHFLAGS          #define 0x00C0 = 0x000000c0  intuition/intuition.h: *125
HIGHIMAGE          #define 0x0000 = 0x00000000  intuition/intuition.h: *126
HIGHITEM           #define 0x2000 = 0x00002000  intuition/intuition.h: *136
HIGHLIGHTTEXTPEN   #define (0x0008) = 0x00000008  intuition/screens.h: *90
HIGHNONE           #define 0x00C0 = 0x000000c0  intuition/intuition.h: *129
HIRES              #define 0x8000 = 0x00008000  graphics/view.h: *101
HIRESDPF2_KEY      #define 0x00008440 = 0x00008440  graphics/displayinfo.h: *170
HIRESDPF_KEY       #define 0x00008400 = 0x00008400  graphics/displayinfo.h: *164
HIRESGADGET        #define 0 = 0x00000000  intuition/intuitionbase.h: *43
HIRESLACEDPF2_KEY #define 0x00008444 = 0x00008444
                   graphics/displayinfo.h: *173
HIRESLACEDPF_KEY   #define 0x00008404 = 0x00008404  graphics/displayinfo.h: *167
HIRESLACE_KEY      #define 0x00008004 = 0x00008004  graphics/displayinfo.h: *160
HIRESPICK          #define 0x0000 = 0x00000000  intuition/intuitionbase.h: *36
HIRES_KEY          #define 0x00008000 = 0x00008000  graphics/displayinfo.h: *156
HOLDNMODIFY        #define 0x800 = 0x00000800  graphics/display.h: *23
HP_LASERJET        #define 0x0B = 0x0000000b  intuition/preferences.h: *204
HP_LASERJET_PLUS   #define 0x0C = 0x0000000c  intuition/preferences.h: *205
HPotRes            unsigned short int in struct PropInfo
           +0x000e intuition/intuition.h: *490
HSIZEBITS          #define 6 = 0x00000006  hardware/blit.h: *15
HSIZEMASK          #define 0x3f = 0x0000003f  hardware/blit.h: *17
HSYNCTRUE          #define 0x0001 = 0x00000001  hardware/custom.h: *159
HUNK_BREAK         #define 1014 = 0x000003f6  dos/doshunks.h: *31
HUNK_BSS           #define 1003 = 0x000003eb  dos/doshunks.h: *20
HUNK_CODE          #define 1001 = 0x000003e9  dos/doshunks.h: *18
HUNK_DATA          #define 1002 = 0x000003ea  dos/doshunks.h: *19
HUNK_DEBUG         #define 1009 = 0x000003f1  dos/doshunks.h: *26
HUNK_DREL16        #define 1016 = 0x000003f8  dos/doshunks.h: *34
HUNK_DREL32        #define 1015 = 0x000003f7  dos/doshunks.h: *33
HUNK_DREL8         #define 1017 = 0x000003f9  dos/doshunks.h: *35
HUNK_END           #define 1010 = 0x000003f2  dos/doshunks.h: *27
HUNK_EXT           #define 1007 = 0x000003ef  dos/doshunks.h: *24
HUNK_HEADER        #define 1011 = 0x000003f3  dos/doshunks.h: *28
HUNK_INDEX         #define 1019 = 0x000003fb  dos/doshunks.h: *38
HUNK_LIB           #define 1018 = 0x000003fa  dos/doshunks.h: *37
HUNK_NAME          #define 1000 = 0x000003e8  dos/doshunks.h: *17
HUNK_OVERLAY       #define 1013 = 0x000003f5  dos/doshunks.h: *30
HUNK_RELOC16       #define 1005 = 0x000003ed  dos/doshunks.h: *22
HUNK_RELOC32       #define 1004 = 0x000003ec  dos/doshunks.h: *21
HUNK_RELOC8        #define 1006 = 0x000003ee  dos/doshunks.h: *23
HUNK_SYMBOL        #define 1008 = 0x000003f0  dos/doshunks.h: *25
HUNK_UNIT          #define 999 = 0x000003e7  dos/doshunks.h: *16
HWAITPOS           #define u3.u4.u2.HWaitPos  graphics/copper.h: *51
HWaitPos           short int in union (no tag)  +0x0000 graphics/copper.h: *41
HeadComp           pointer to struct AnimComp in struct AnimOb
           +0x0024 graphics/gels.h: *227
HeadOb             pointer to struct AnimOb in struct AnimComp
           +0x001e graphics/gels.h: *200
Header             struct QueryHeader(size 0x0010 bytes) in struct DisplayInfo
           +0x0000 graphics/displayinfo.h: *52
Header             struct QueryHeader(size 0x0010 bytes) in struct DimensionInfo
           +0x0000 graphics/displayinfo.h: *94
Header             struct QueryHeader(size 0x0010 bytes) in struct MonitorInfo
           +0x0000 graphics/displayinfo.h: *111
```

```
Header             struct QueryHeader(size 0x0010 bytes) in struct NameInfo
           +0x0000 graphics/displayinfo.h: *134
Height             short int in struct Layer +0x0088 graphics/clip.h: *57
Height             short int in struct Menu +0x000a intuition/intuition.h: *66
Height             short int in struct MenuItem
           +0x000a intuition/intuition.h: *94
Height             short int in struct Requester
           +0x000a intuition/intuition.h: *150
Height             short int in struct Gadget +0x000a intuition/intuition.h: *221
Height             short int in struct Image +0x0006 intuition/intuition.h: *624
Height             short int in struct IBox +0x0006 intuition/intuition.h: *787
Height             short int in struct Window +0x000a intuition/intuition.h: *800
Height             short int in struct NewWindow
           +0x0006 intuition/intuition.h: *977
Height             short int in struct ExtNewWindow
           +0x0006 intuition/intuition.h: *1047
Height             short int in struct Screen +0x000e intuition/screens.h: *104
Height             short int in struct NewScreen
           +0x0006 intuition/screens.h: *312
Height             short int in struct ExtNewScreen
           +0x0006 intuition/screens.h: *348
Height             short int in struct VSprite +0x001a graphics/gels.h: *103
Height             short int in struct (no tag)
           +0x0002 intuition/imageclass.h: *166
Height             short int in struct (no tag)
           +0x0002 intuition/imageclass.h: *183
Height             short int in struct (no tag)
           +0x0002 intuition/imageclass.h: *198
HitMask            short int in struct VSprite +0x0022 graphics/gels.h: *108
Hook               structure tag size 0x0014 utility/hooks.h: *24
                   graphics/clip.h: 53
                   dos/exall.h: 67
                   intuition/classes.h: 29
                   intuition/sghooks.h: 27
HorizBody          unsigned short int in struct PropInfo
           +0x0006 intuition/intuition.h: *484
HorizPot           unsigned short int in struct PropInfo
           +0x0002 intuition/intuition.h: *464
id_BytesPerBlock long int in struct InfoData +0x0014 dos/dos.h: *125
id_DiskState       long int in struct InfoData +0x0008 dos/dos.h: *122
id_DiskType        long int in struct InfoData +0x0018 dos/dos.h: *126
id_InUse           long int in struct InfoData +0x0020 dos/dos.h: *128
id_NumBlocks       long int in struct InfoData +0x000c dos/dos.h: *123
id_NumBlocksUsed long int in struct InfoData +0x0010 dos/dos.h: *124
id_NumSoftErrors long int in struct InfoData +0x0000 dos/dos.h: *120
id_UnitNumber      long int in struct InfoData +0x0004 dos/dos.h: *121
id_VolumeNode      long int in struct InfoData +0x001c dos/dos.h: *127
ie_Class           unsigned char in struct InputEvent
           +0x0004 devices/inputevent.h: *201
ie_Code            unsigned short int in struct InputEvent
           +0x0006 devices/inputevent.h: *203
ie_EventAddress    #define ie_position.ie_addr  devices/inputevent.h: *223
ie_NextEvent       pointer to struct InputEvent in struct InputEvent
           +0x0000 devices/inputevent.h: *200
ie_Prev1DownCode #define ie_position.ie_dead.ie_prev1DownCode
                   devices/inputevent.h: *224
ie_Prev1DownQual #define ie_position.ie_dead.ie_prev1DownQual
                   devices/inputevent.h: *225
ie_Prev2DownCode #define ie_position.ie_dead.ie_prev2DownCode
                   devices/inputevent.h: *226
ie_Prev2DownQual #define ie_position.ie_dead.ie_prev2DownQual
                   devices/inputevent.h: *227
ie_Qualifier       unsigned short int in struct InputEvent
           +0x0008 devices/inputevent.h: *204
ie_SubClass        unsigned char in struct InputEvent
           +0x0005 devices/inputevent.h: *202
```

Include File Cross Reference Page 65

```
ie_TimeStamp     struct timeval(size 0x0008 bytes) in struct InputEvent
         +0x000e devices/inputevent.h: *218
ie_X             #define ie_position.ie_xy.ie_x  devices/inputevent.h: *221
ie_Y             #define ie_position.ie_xy.ie_y  devices/inputevent.h: *222
ie_addr          pointer to void in union (no tag)
         +0x0000 devices/inputevent.h: *210
ie_dead          struct (no tag) (size 0x0004 bytes) in union (no tag)
         +0x0000 devices/inputevent.h: *216
ie_position      union (no tag) (size 0x0004 bytes) in struct InputEvent
         +0x000a devices/inputevent.h: *217
ie_prev1DownCode unsigned char in struct (no tag)
         +0x0000 devices/inputevent.h: *212
ie_prev1DownQual unsigned char in struct (no tag)
         +0x0001 devices/inputevent.h: *213
ie_prev2DownCode unsigned char in struct (no tag)
         +0x0002 devices/inputevent.h: *214
ie_prev2DownQual unsigned char in struct (no tag)
         +0x0003 devices/inputevent.h: *215
ie_x             short int in struct (no tag)
         +0x0000 devices/inputevent.h: *207
ie_xy            struct (no tag) (size 0x0004 bytes) in union (no tag)
         +0x0000 devices/inputevent.h: *209
ie_y             short int in struct (no tag)
         +0x0002 devices/inputevent.h: *208
iepp_Position    struct (no tag) (size 0x0004 bytes) in struct IEPointerPixel
         +0x0004 devices/inputevent.h: *93
iepp_Screen      pointer to struct Screen in struct IEPointerPixel
         +0x0000 devices/inputevent.h: *89
iept_Pressure    short int in struct IEPointerTablet
         +0x0008 devices/inputevent.h: *118
iept_Range       struct (no tag) (size 0x0004 bytes) in struct IEPointerTablet
         +0x0000 devices/inputevent.h: *112
iept_Value       struct (no tag) (size 0x0004 bytes) in struct IEPointerTablet
         +0x0004 devices/inputevent.h: *116
iff_Depth        long int in struct IFFHandle +0x0008 libraries/iffparse.h: *38
iff_Flags        unsigned long int in struct IFFHandle
         +0x0004 libraries/iffparse.h: *37
iff_Stream       unsigned long int in struct IFFHandle
         +0x0000 libraries/iffparse.h: *36
impDraw          structure tag size 0x0018 intuition/imageclass.h: *152
impErase         structure tag (size 0x0010 bytes)in struct impErase
                 intuition/imageclass.h: *172
impFrameBox      structure tag size 0x0014 intuition/imageclass.h: *139
impHitTest       structure tag (size 0x000c bytes)in struct impHitTest
                 intuition/imageclass.h: *188
imp_ContentsBox pointer to struct IBox in struct impFrameBox
         +0x0004 intuition/imageclass.h: *141
imp_Dimensions   struct (no tag) (size 0x0004 bytes) in struct impDraw
         +0x0014 intuition/imageclass.h: *167
imp_Dimensions   struct (no tag) (size 0x0004 bytes) in struct impErase
         +0x000c intuition/imageclass.h: *184
imp_Dimensions   struct (no tag) (size 0x0004 bytes) in struct impHitTest
         +0x0008 intuition/imageclass.h: *199
imp_DrInfo       pointer to struct DrawInfo in struct impFrameBox
         +0x000c intuition/imageclass.h: *143
imp_DrInfo       pointer to struct DrawInfo in struct impDraw
         +0x0010 intuition/imageclass.h: *161
imp_FrameBox     pointer to struct IBox in struct impFrameBox
         +0x0008 intuition/imageclass.h: *142
imp_FrameFlags   unsigned long int in struct impFrameBox
         +0x0010 intuition/imageclass.h: *144
imp_Offset       struct (no tag) (size 0x0004 bytes) in struct impDraw
         +0x0008 intuition/imageclass.h: *158
imp_Offset       struct (no tag) (size 0x0004 bytes) in struct impErase
         +0x0008 intuition/imageclass.h: *178
imp_Point        struct (no tag) (size 0x0004 bytes) in struct impHitTest
```

Include File Cross Reference Page 66

```
         +0x0004 intuition/imageclass.h: *193
imp_RPort        pointer to struct RastPort in struct impDraw
         +0x0004 intuition/imageclass.h: *154
imp_RPort        pointer to struct RastPort in struct impErase
         +0x0004 intuition/imageclass.h: *174
imp_State        unsigned long int in struct impDraw
         +0x000c intuition/imageclass.h: *160
intena           unsigned short int in struct Custom
         +0x009a hardware/custom.h: *94
intenar          unsigned short int in struct Custom
         +0x001c hardware/custom.h: *42
intreq           unsigned short int in struct Custom
         +0x009c hardware/custom.h: *95
intreqr          unsigned short int in struct Custom
         +0x001e hardware/custom.h: *43
io_Actual        unsigned long int in struct IOStdReq +0x0020 exec/io.h: *36
io_Actual        unsigned long int in struct IOClipReq
         +0x0020 devices/clipboard.h: *50
io_Baud          unsigned long int in struct IOExtSer
         +0x003c devices/serial.h: *65
io_BrkTime       unsigned long int in struct IOExtSer
         +0x0040 devices/serial.h: *66
io_ClipID        long int in struct IOClipReq +0x0030 devices/clipboard.h: *54
io_ColorMap      pointer to struct ColorMap in struct IODRPReq
         +0x0024 devices/printer.h: *164
io_Command       unsigned short int in struct IORequest +0x001c exec/io.h: *24
io_Command       unsigned short int in struct IOStdReq +0x001c exec/io.h: *33
io_Command       unsigned short int in struct IOClipReq
         +0x001c devices/clipboard.h: *47
io_Command       unsigned short int in struct IOPrtCmdReq
         +0x001c devices/printer.h: *146
io_Command       unsigned short int in struct IODRPReq
         +0x001c devices/printer.h: *160
io_CtlChar       unsigned long int in struct IOExtSer
         +0x0030 devices/serial.h: *62
io_Data          pointer to void in struct IOStdReq +0x0028 exec/io.h: *38
io_Data          pointer to unsigned char in struct IOClipReq
         +0x0028 devices/clipboard.h: *52
io_DestCols      long int in struct IODRPReq +0x0034 devices/printer.h: *170
io_DestRows      long int in struct IODRPReq +0x0038 devices/printer.h: *171
io_Device        pointer to struct Device in struct IORequest
         +0x0014 exec/io.h: *22
io_Device        pointer to struct Device in struct IOStdReq
         +0x0014 exec/io.h: *31
io_Device        pointer to struct Device in struct IOClipReq
         +0x0014 devices/clipboard.h: *45
io_Device        pointer to struct Device in struct IOPrtCmdReq
         +0x0014 devices/printer.h: *144
io_Device        pointer to struct Device in struct IODRPReq
         +0x0014 devices/printer.h: *158
io_Error         char in struct IORequest +0x001f exec/io.h: *26
io_Error         char in struct IOStdReq +0x001f exec/io.h: *35
io_Error         char in struct IOClipReq +0x001f devices/clipboard.h: *49
io_Error         char in struct IOPrtCmdReq +0x001f devices/printer.h: *148
io_Error         char in struct IODRPReq +0x001f devices/printer.h: *162
io_ExtFlags      unsigned long int in struct IOExtSer
         +0x0038 devices/serial.h: *64
io_Flags         unsigned char in struct IORequest +0x001e exec/io.h: *25
io_Flags         unsigned char in struct IOStdReq +0x001e exec/io.h: *34
io_Flags         unsigned char in struct IOClipReq
         +0x001e devices/clipboard.h: *48
io_Flags         unsigned char in struct IOPrtCmdReq
         +0x001e devices/printer.h: *147
io_Flags         unsigned char in struct IODRPReq
         +0x001e devices/printer.h: *161
io_Length        unsigned long int in struct IOStdReq +0x0024 exec/io.h: *37
```

Include File Cross Reference Page 67

```
io_Length         unsigned long int in struct IOClipReq
          +0x0024 devices/clipboard.h: *51
io_Message        struct Message(size 0x0014 bytes) in struct IORequest
          +0x0000 exec/io.h: *21
io_Message        struct Message(size 0x0014 bytes) in struct IOStdReq
          +0x0000 exec/io.h: *30
io_Message        struct Message(size 0x0014 bytes) in struct IOClipReq
          +0x0000 devices/clipboard.h: *44
io_Message        struct Message(size 0x0014 bytes) in struct IOPrtCmdReq
          +0x0000 devices/printer.h: *143
io_Message        struct Message(size 0x0014 bytes) in struct IODRPReq
          +0x0000 devices/printer.h: *157
io_Modes          unsigned long int in struct IODRPReq
          +0x0028 devices/printer.h: *165
io_Offset         unsigned long int in struct IOStdReq +0x002c exec/io.h: *39
io_Offset         unsigned long int in struct IOClipReq
          +0x002c devices/clipboard.h: *53
io_PExtFlags      unsigned long int in struct IOExtPar
          +0x0030 devices/parallel.h: *52
io_PTermArray     struct IOPArray(size 0x0008 bytes) in struct IOExtPar
          +0x0036 devices/parallel.h: *55
io_ParFlags       unsigned char in struct IOExtPar
          +0x0035 devices/parallel.h: *54
io_Parm0          unsigned char in struct IOPrtCmdReq
          +0x0022 devices/printer.h: *150
io_Parm1          unsigned char in struct IOPrtCmdReq
          +0x0023 devices/printer.h: *151
io_Parm2          unsigned char in struct IOPrtCmdReq
          +0x0024 devices/printer.h: *152
io_Parm3          unsigned char in struct IOPrtCmdReq
          +0x0025 devices/printer.h: *153
io_PrtCommand     unsigned short int in struct IOPrtCmdReq
          +0x0020 devices/printer.h: *149
io_RBufLen        unsigned long int in struct IOExtSer
          +0x0034 devices/serial.h: *63
io_RastPort       pointer to struct RastPort in struct IODRPReq
          +0x0020 devices/printer.h: *163
io_ReadLen        unsigned char in struct IOExtSer +0x004c devices/serial.h: *68
io_SerFlags       unsigned char in struct IOExtSer +0x004f devices/serial.h: *71
io_Special        unsigned short int in struct IODRPReq
          +0x003c devices/printer.h: *172
io_SrcHeight      unsigned short int in struct IODRPReq
          +0x0032 devices/printer.h: *169
io_SrcWidth       unsigned short int in struct IODRPReq
          +0x0030 devices/printer.h: *168
io_SrcX           unsigned short int in struct IODRPReq
          +0x002c devices/printer.h: *166
io_SrcY           unsigned short int in struct IODRPReq
          +0x002e devices/printer.h: *167
io_Status         unsigned char in struct IOExtPar
          +0x0034 devices/parallel.h: *53
io_Status         unsigned short int in struct IOExtSer
          +0x0050 devices/serial.h: *72
io_StopBits       unsigned char in struct IOExtSer +0x004e devices/serial.h: *70
io_TermArray      struct IOTArray(size 0x0008 bytes) in struct IOExtSer
          +0x0044 devices/serial.h: *67
io_Unit           pointer to struct Unit in struct IORequest
          +0x0018 exec/io.h: *23
io_Unit           pointer to struct Unit in struct IOStdReq
          +0x0018 exec/io.h: *32
io_Unit           pointer to struct ClipboardUnitPartial in struct IOClipReq
          +0x0018 devices/clipboard.h: *46
io_Unit           pointer to struct Unit in struct IOPrtCmdReq
          +0x0018 devices/printer.h: *145
io_Unit           pointer to struct Unit in struct IODRPReq
          +0x0018 devices/printer.h: *159
```

Include File Cross Reference Page 68

```
io_WriteLen       unsigned char in struct IOExtSer +0x004d devices/serial.h: *69
ioa_AllocKey      short int in struct IOAudio +0x0020 devices/audio.h: *49
ioa_Cycles        unsigned short int in struct IOAudio
          +0x002e devices/audio.h: *54
ioa_Data          pointer to unsigned char in struct IOAudio
          +0x0022 devices/audio.h: *50
ioa_Length        unsigned long int in struct IOAudio
          +0x0026 devices/audio.h: *51
ioa_Period        unsigned short int in struct IOAudio
          +0x002a devices/audio.h: *52
ioa_Request       struct IORequest(size 0x0020 bytes) in struct IOAudio
          +0x0000 devices/audio.h: *48
ioa_Volume        unsigned short int in struct IOAudio
          +0x002c devices/audio.h: *53
ioa_WriteMsg      struct Message(size 0x0014 bytes) in struct IOAudio
          +0x0030 devices/audio.h: *55
iobArea           array [204] of char in struct IoBuff
          +0x0034 rexx/rexxio.h: *32
iobBct            long int in struct IoBuff +0x0030 rexx/rexxio.h: *31
iobDFH            long int in struct IoBuff +0x0028 rexx/rexxio.h: *29
iobLock           pointer to void in struct IoBuff +0x002c rexx/rexxio.h: *30
iobNode           struct RexxRsrc(size 0x0020 bytes) in struct IoBuff
          +0x0000 rexx/rexxio.h: *26
iobRct            long int in struct IoBuff +0x0024 rexx/rexxio.h: *28
iobRpt            pointer to void in struct IoBuff +0x0020 rexx/rexxio.h: *27
iotd_Count        unsigned long int in struct IOExtTD
          +0x0030 devices/trackdisk.h: *121
iotd_Req          struct IOStdReq(size 0x0030 bytes) in struct IOExtTD
          +0x0000 devices/trackdisk.h: *120
iotd_SecLabel     unsigned long int in struct IOExtTD
          +0x0034 devices/trackdisk.h: *122
is_Code           pointer to function returning void in struct Interrupt
          +0x0012 exec/interrupts.h: *27
is_Data           pointer to void in struct Interrupt
          +0x000e exec/interrupts.h: *26
is_Node           struct Node(size 0x000e bytes) in struct Interrupt
          +0x0000 exec/interrupts.h: *25
is_Node           struct Node(size 0x000e bytes) in struct Isrvstr
          +0x0000 graphics/graphint.h: *22
itof              macro (1 argument)   libraries/mathffp.h: *33
                  libraries/mathieeedp.h: *33
iv_Code           pointer to function returning void in struct IntVector
          +0x0004 exec/interrupts.h: *33
iv_Data           pointer to void in struct IntVector
          +0x0000 exec/interrupts.h: *32
iv_Node           pointer to struct Node in struct IntVector
          +0x0008 exec/interrupts.h: *34
ix_Class          unsigned char in struct InputXpression
          +0x0001 libraries/commodities.h: *175
ix_Code           unsigned short int in struct InputXpression
          +0x0002 libraries/commodities.h: *177
ix_CodeMask       unsigned short int in struct InputXpression
          +0x0004 libraries/commodities.h: *179
ix_QualMask       unsigned short int in struct InputXpression
          +0x0008 libraries/commodities.h: *185
ix_QualSame       unsigned short int in struct InputXpression
          +0x000a libraries/commodities.h: *189
ix_Qualifier      unsigned short int in struct InputXpression
          +0x0006 libraries/commodities.h: *183
ix_Version        unsigned char in struct InputXpression
          +0x0000 libraries/commodities.h: *174
IA_APATSIZE       #define IA_APatSize = 0x80020011  intuition/iobsolete.h: *246
IA_APATTERN       #define IA_APattern = 0x80020010  intuition/iobsolete.h: *245
IA_APatSize       #define (IA_Dummy + 0x11) = 0x80020011
                  intuition/imageclass.h: *63
IA_APattern       #define (IA_Dummy + 0x10) = 0x80020010
```

Include File Cross Reference Page 69

```
                 intuition/imageclass.h: *62
IA_BGPEN         #define IA_BGPen = 0x80020006  intuition/iobsolete.h: *240
IA_BGPen         #define (IA_Dummy + 0x06) = 0x80020006
                 intuition/imageclass.h: *41
IA_DATA          #define IA_Data = 0x80020007  intuition/iobsolete.h: *241
IA_DOUBLEEMBOSS  #define IA_DoubleEmboss = 0x80020016
                 intuition/iobsolete.h: *251
IA_Data          #define (IA_Dummy + 0x07) = 0x80020007
                 intuition/imageclass.h: *43
IA_DoubleEmboss  #define (IA_Dummy + 0x16) = 0x80020016
                 intuition/imageclass.h: *68
IA_Dummy         #define (TAG_USER + 0x20000) = 0x80020000
                 intuition/imageclass.h: *34
IA_EDGESONLY     #define IA_EdgesOnly = 0x80020017
                 intuition/iobsolete.h: *252
IA_EdgesOnly     #define (IA_Dummy + 0x17) = 0x80020017
                 intuition/imageclass.h: *69
IA_FGPEN         #define IA_FGPen = 0x80020005  intuition/iobsolete.h: *239
IA_FGPen         #define (IA_Dummy + 0x05) = 0x80020005
                 intuition/imageclass.h: *39
IA_FONT          #define IA_Font = 0x80020013  intuition/iobsolete.h: *248
IA_Font          #define (IA_Dummy + 0x13) = 0x80020013
                 intuition/imageclass.h: *65
IA_HEIGHT        #define IA_Height = 0x80020004  intuition/iobsolete.h: *238
IA_HIGHLIGHTPEN  #define IA_HighlightPen = 0x8002000a
                 intuition/iobsolete.h: *254
IA_Height        #define (IA_Dummy + 0x04) = 0x80020004
                 intuition/imageclass.h: *38
IA_HighlightPen  #define (IA_Dummy + 0x0A) = 0x8002000a
                 intuition/imageclass.h: *86
IA_LEFT          #define IA_Left = 0x80020001  intuition/iobsolete.h: *235
IA_LINEWIDTH     #define IA_LineWidth = 0x80020008
                 intuition/iobsolete.h: *242
IA_Left          #define (IA_Dummy + 0x01) = 0x80020001
                 intuition/imageclass.h: *35
IA_LineWidth     #define (IA_Dummy + 0x08) = 0x80020008
                 intuition/imageclass.h: *47
IA_MODE          #define IA_Mode = 0x80020012  intuition/iobsolete.h: *247
IA_Mode          #define (IA_Dummy + 0x12) = 0x80020012
                 intuition/imageclass.h: *64
IA_OUTLINE       #define IA_Outline = 0x80020014  intuition/iobsolete.h: *249
IA_Outline       #define (IA_Dummy + 0x14) = 0x80020014
                 intuition/imageclass.h: *66
IA_PENS          #define IA_Pens = 0x8002000e  intuition/iobsolete.h: *243
IA_Pens          #define (IA_Dummy + 0x0E) = 0x8002000e
                 intuition/imageclass.h: *48
IA_RECESSED      #define IA_Recessed = 0x80020015  intuition/iobsolete.h: *250
IA_RESOLUTION    #define IA_Resolution = 0x8002000f
                 intuition/iobsolete.h: *244
IA_Recessed      #define (IA_Dummy + 0x15) = 0x80020015
                 intuition/imageclass.h: *67
IA_Resolution    #define (IA_Dummy + 0x0F) = 0x8002000f
                 intuition/imageclass.h: *55
IA_SHADOWPEN     #define IA_ShadowPen = 0x80020009
                 intuition/iobsolete.h: *253
IA_ShadowPen     #define (IA_Dummy + 0x09) = 0x80020009
                 intuition/imageclass.h: *85
IA_TOP           #define IA_Top = 0x80020002  intuition/iobsolete.h: *236
IA_Top           #define (IA_Dummy + 0x02) = 0x80020002
                 intuition/imageclass.h: *36
IA_WIDTH         #define IA_Width = 0x80020003  intuition/iobsolete.h: *237
IA_Width         #define (IA_Dummy + 0x03) = 0x80020003
                 intuition/imageclass.h: *37
IAddress         pointer to void in struct IntuiMessage
        +0x001c  intuition/intuition.h: *695
IBox             structure tag size 0x0008 intuition/intuition.h: *783
```

Include File Cross Reference Page 70

```
                 intuition/cghooks.h: 49, 72, 73
                 intuition/imageclass.h: 141, 142
ICA_Dummy        #define (TAG_USER+0x40000L) = 0x80040000
                 intuition/icclass.h: *28
ICA_MAP          #define (ICA_Dummy + 2) = 0x80040002
                 intuition/icclass.h: *31
ICA_TARGET       #define (ICA_Dummy + 1) = 0x80040001
                 intuition/icclass.h: *29
ICCLASS          #define "icclass"  intuition/classusr.h: *53
ICM_CHECKLOOP    #define (0x0404L) = 0x00000404  intuition/icclass.h: *23
ICM_CLEARLOOP    #define (0x0403L) = 0x00000403  intuition/icclass.h: *22
ICM_Dummy        #define (0x0401L) = 0x00000401  intuition/icclass.h: *20
ICM_SETLOOP      #define (0x0402L) = 0x00000402  intuition/icclass.h: *21
ICONNAME         #define "icon.library"  workbench/icon.h: *15
ICSPECIAL_CODE   #define (ICA_Dummy + 3) = 0x80040003
                 intuition/icclass.h: *33
ICTARGET_IDCMP   #define (~0L) = 0xffffffff  intuition/icclass.h: *48
IClass           structure tag size 0x0034 intuition/classes.h: *28, 31, 70
IDCMPFlags       unsigned long int in struct Window
        +0x0052  intuition/intuition.h: *855
IDCMPFlags       unsigned long int in struct NewWindow
        +0x000a  intuition/intuition.h: *981
IDCMPFlags       unsigned long int in struct ExtNewWindow
        +0x000a  intuition/intuition.h: *1050
IDCMPUPDATE      #define IDCMP_IDCMPUPDATE = 0x00800000
                 intuition/iobsolete.h: *137
IDCMPWindow      pointer to struct Window in struct IntuiMessage
        +0x002c  intuition/intuition.h: *713
IDCMP_ACTIVEWINDOW #define 0x00040000 = 0x00040000
                 intuition/intuition.h: *742
IDCMP_CHANGEWINDOW #define 0x02000000 = 0x02000000
                 intuition/intuition.h: *752
IDCMP_CLOSEWINDOW #define 0x00000200 = 0x00000200  intuition/intuition.h: *733
IDCMP_DELTAMOVE  #define 0x00100000 = 0x00100000  intuition/intuition.h: *744
IDCMP_DISKINSERTED #define 0x00008000 = 0x00008000
                 intuition/intuition.h: *739
IDCMP_DISKREMOVED #define 0x00010000 = 0x00010000  intuition/intuition.h: *740
IDCMP_GADGETDOWN #define 0x00000020 = 0x00000020  intuition/intuition.h: *729
IDCMP_GADGETUP   #define 0x00000040 = 0x00000040  intuition/intuition.h: *730
IDCMP_IDCMPUPDATE #define 0x00800000 = 0x00800000  intuition/intuition.h: *748
IDCMP_INACTIVEWINDOW #define 0x00080000 = 0x00080000
                 intuition/intuition.h: *743
IDCMP_INTUITICKS #define 0x00400000 = 0x00400000  intuition/intuition.h: *746
IDCMP_LONELYMESSAGE #define 0x80000000 = 0x80000000
                 intuition/intuition.h: *761
IDCMP_MENUHELP   #define 0x01000000 = 0x01000000  intuition/intuition.h: *750
IDCMP_MENUPICK   #define 0x00000100 = 0x00000100  intuition/intuition.h: *732
IDCMP_MENUVERIFY #define 0x00002000 = 0x00002000  intuition/intuition.h: *737
IDCMP_MOUSEBUTTONS #define 0x00000008 = 0x00000008
                 intuition/intuition.h: *727
IDCMP_MOUSEMOVE  #define 0x00000010 = 0x00000010  intuition/intuition.h: *728
IDCMP_NEWPREFS   #define 0x00004000 = 0x00004000  intuition/intuition.h: *738
IDCMP_NEWSIZE    #define 0x00000002 = 0x00000002  intuition/intuition.h: *725
IDCMP_RAWKEY     #define 0x00000400 = 0x00000400  intuition/intuition.h: *734
IDCMP_REFRESHWINDOW #define 0x00000004 = 0x00000004
                 intuition/intuition.h: *726
IDCMP_REQCLEAR   #define 0x00001000 = 0x00001000  intuition/intuition.h: *736
IDCMP_REQSET     #define 0x00000080 = 0x00000080  intuition/intuition.h: *731
IDCMP_REQVERIFY  #define 0x00000800 = 0x00000800  intuition/intuition.h: *735
IDCMP_SIZEVERIFY #define 0x00000001 = 0x00000001  intuition/intuition.h: *724
IDCMP_VANILLAKEY #define 0x00200000 = 0x00200000  intuition/intuition.h: *745
IDCMP_WBENCHMESSAGE #define 0x00020000 = 0x00020000
                 intuition/intuition.h: *741
IDNAME_BADBLOCK  #define 0x42414442 = 0x42414442  devices/hardblocks.h: *129
IDNAME_FILESYSHEADER #define 0x46534844 = 0x46534844
                 devices/hardblocks.h: *186
```

```
IDNAME_LOADSEG   #define 0x4C534547 = 0x4c534547  devices/hardblocks.h: *199
IDNAME_PARTITION #define 0x50415254 = 0x50415254  devices/hardblocks.h: *148
IDNAME_RIGIDDISK #define 0x5244534B = 0x5244534b  devices/hardblocks.h: *95
IDNestCnt        char in struct ExecBase +0x0126 exec/execbase.h: *70
IDS_BUSY         #define (3L) = 0x00000003  intuition/imageclass.h: *129
IDS_DISABLED     #define (2L) = 0x00000002  intuition/imageclass.h: *128
IDS_INACTIVEDISABLED #define (7L) = 0x00000007  intuition/imageclass.h: *133
IDS_INACTIVENORMAL #define (5L) = 0x00000005  intuition/imageclass.h: *131
IDS_INACTIVESELECTED #define (6L) = 0x00000006  intuition/imageclass.h: *132
IDS_INDETERMINANT #define IDS_INDETERMINATE = 0x00000004
                 intuition/imageclass.h: *136
IDS_INDETERMINATE #define (4L) = 0x00000004  intuition/imageclass.h: *130
IDS_NORMAL       #define (0L) = 0x00000000  intuition/imageclass.h: *126
IDS_SELECTED     #define (1L) = 0x00000001  intuition/imageclass.h: *127
ID_CAT           #define MAKE_ID('C','A','T',' ') = 0x43415420
                 libraries/iffparse.h: *150
ID_DOS_DISK      #define (0x444F5300L) = 0x444f5300  dos/dos.h: *140
ID_FFS_DISK      #define (0x444F5301L) = 0x444f5301  dos/dos.h: *141
ID_FORM          #define MAKE_ID('F','O','R','M') = 0x464f524d
                 libraries/iffparse.h: *148
ID_KICKSTART_DISK #define (0x4B49434BL) = 0x4b49434b  dos/dos.h: *143
ID_LIST          #define MAKE_ID('L','I','S','T') = 0x4c495354
                 libraries/iffparse.h: *149
ID_MSDOS_DISK    #define (0x4d534400L) = 0x4d534400  dos/dos.h: *144
ID_NOT_REALLY_DOS #define (0x4E444F53L) = 0x4e444f53  dos/dos.h: *142
ID_NO_DISK_PRESENT #define (-1) = 0xffffffff  dos/dos.h: *138
ID_NULL          #define MAKE_ID(' ',' ',' ',' ') = 0x20202020
                 libraries/iffparse.h: *152
ID_PROP          #define MAKE_ID('P','R','O','P') = 0x50524f50
                 libraries/iffparse.h: *151
ID_UNREADABLE_DISK #define (0x42414400L) = 0x42414400  dos/dos.h: *139
ID_VALIDATED     #define 82 = 0x00000052  dos/dos.h: *135
ID_VALIDATING    #define 81 = 0x00000051  dos/dos.h: *134
ID_WRITE_PROTECTED #define 80 = 0x00000050  dos/dos.h: *133
IECLASS_ACTIVEWINDOW #define 0x11 = 0x00000011  devices/inputevent.h: *55
IECLASS_CHANGEWINDOW #define 0x15 = 0x00000015  devices/inputevent.h: *63
IECLASS_CLOSEWINDOW #define 0x0B = 0x0000000b  devices/inputevent.h: *43
IECLASS_DISKINSERTED #define 0x10 = 0x00000010  devices/inputevent.h: *53
IECLASS_DISKREMOVED #define 0x0F = 0x0000000f  devices/inputevent.h: *51
IECLASS_EVENT    #define 0x03 = 0x00000003  devices/inputevent.h: *29
IECLASS_GADGETDOWN #define 0x07 = 0x00000007  devices/inputevent.h: *35
IECLASS_GADGETUP #define 0x08 = 0x00000008  devices/inputevent.h: *37
IECLASS_INACTIVEWINDOW #define 0x12 = 0x00000012  devices/inputevent.h: *57
IECLASS_MAX      #define 0x15 = 0x00000015  devices/inputevent.h: *66
                 devices/conunit.h: 99
IECLASS_MENUHELP #define 0x14 = 0x00000014  devices/inputevent.h: *61
IECLASS_MENULIST #define 0x0A = 0x0000000a  devices/inputevent.h: *41
IECLASS_NEWPOINTERPOS #define 0x13 = 0x00000013  devices/inputevent.h: *59
IECLASS_NEWPREFS #define 0x0E = 0x0000000e  devices/inputevent.h: *49
IECLASS_NULL     #define 0x00 = 0x00000000  devices/inputevent.h: *23
IECLASS_POINTERPOS #define 0x04 = 0x00000004  devices/inputevent.h: *31
IECLASS_RAWKEY   #define 0x01 = 0x00000001  devices/inputevent.h: *25
IECLASS_RAWMOUSE #define 0x02 = 0x00000002  devices/inputevent.h: *27
IECLASS_REFRESHWINDOW #define 0x0D = 0x0000000d  devices/inputevent.h: *47
IECLASS_REQUESTER #define 0x09 = 0x00000009  devices/inputevent.h: *39
IECLASS_SIZEWINDOW #define 0x0C = 0x0000000c  devices/inputevent.h: *45
IECLASS_TIMER    #define 0x06 = 0x00000006  devices/inputevent.h: *33
IECODE_ASCII_DEL #define 0x7F = 0x0000007f  devices/inputevent.h: *136
IECODE_ASCII_FIRST #define 0x20 = 0x00000020  devices/inputevent.h: *134
IECODE_ASCII_LAST #define 0x7E = 0x0000007e  devices/inputevent.h: *135
IECODE_C0_FIRST  #define 0x00 = 0x00000000  devices/inputevent.h: *132
IECODE_C0_LAST   #define 0x1F = 0x0000001f  devices/inputevent.h: *133
IECODE_C1_FIRST  #define 0x80 = 0x00000080  devices/inputevent.h: *137
IECODE_C1_LAST   #define 0x9F = 0x0000009f  devices/inputevent.h: *138
IECODE_COMM_CODE_FIRST #define 0x78 = 0x00000078  devices/inputevent.h: *128
IECODE_COMM_CODE_LAST #define 0x7F = 0x0000007f  devices/inputevent.h: *129
```

```
IECODE_KEY_CODE_FIRST #define 0x00 = 0x00000000  devices/inputevent.h: *126
IECODE_KEY_CODE_LAST #define 0x77 = 0x00000077  devices/inputevent.h: *127
IECODE_LATIN1_FIRST #define 0xA0 = 0x000000a0  devices/inputevent.h: *139
IECODE_LATIN1_LAST #define 0xFF = 0x000000ff  devices/inputevent.h: *140
IECODE_LBUTTON   #define 0x68 = 0x00000068  devices/inputevent.h: *143
IECODE_MBUTTON   #define 0x6A = 0x0000006a  devices/inputevent.h: *145
IECODE_NEWACTIVE #define 0x01 = 0x00000001  devices/inputevent.h: *149
IECODE_NEWSIZE   #define 0x02 = 0x00000002  devices/inputevent.h: *150
IECODE_NOBUTTON  #define 0xFF = 0x000000ff  devices/inputevent.h: *146
IECODE_RBUTTON   #define 0x69 = 0x00000069  devices/inputevent.h: *144
IECODE_REFRESH   #define 0x03 = 0x00000003  devices/inputevent.h: *151
IECODE_REQCLEAR  #define 0x00 = 0x00000000  devices/inputevent.h: *158
IECODE_REQSET    #define 0x01 = 0x00000001  devices/inputevent.h: *156
IECODE_UP_PREFIX #define 0x80 = 0x00000080  devices/inputevent.h: *125
IEEEDPACos       function returning "LONG"   libraries/mathieeedp.h: *57
IEEEDPASin       function returning "LONG"   libraries/mathieeedp.h: *58
IEEEDPAbs        function returning "LONG"   libraries/mathieeedp.h: *70
IEEEDPAdd        function returning "LONG"   libraries/mathieeedp.h: *72
IEEEDPAtan       function returning "LONG"   libraries/mathieeedp.h: *56
IEEEDPCeil       function returning "LONG"   libraries/mathieeedp.h: *77
IEEEDPCos        function returning "LONG"   libraries/mathieeedp.h: *57
IEEEDPCosh       function returning "LONG"   libraries/mathieeedp.h: *63
IEEEDPDiv        function returning "LONG"   libraries/mathieeedp.h: *75
IEEEDPExp        function returning "LONG"   libraries/mathieeedp.h: *59
IEEEDPFieee      function returning "LONG"   libraries/mathieeedp.h: *65
IEEEDPFloor      function returning "LONG"   libraries/mathieeedp.h: *76
IEEEDPFlt        function returning "LONG"   libraries/mathieeedp.h: *69
IEEEDPLog        function returning "LONG"   libraries/mathieeedp.h: *59
IEEEDPLog10      function returning "LONG"   libraries/mathieeedp.h: *61
IEEEDPMul        function returning "LONG"   libraries/mathieeedp.h: *74
IEEEDPNeg        function returning "LONG"   libraries/mathieeedp.h: *71
IEEEDPPow        function returning "LONG"   libraries/mathieeedp.h: *61
IEEEDPSin        function returning "LONG"   libraries/mathieeedp.h: *58
IEEEDPSincos     function returning "LONG"   libraries/mathieeedp.h: *62
IEEEDPSinh       function returning "LONG"   libraries/mathieeedp.h: *63
IEEEDPSqrt       function returning "LONG"   libraries/mathieeedp.h: *60
IEEEDPSub        function returning "LONG"   libraries/mathieeedp.h: *73
IEEEDPTan        function returning "LONG"   libraries/mathieeedp.h: *56
IEEEDPTanh       function returning "LONG"   libraries/mathieeedp.h: *63
IEEEDPTieee      function returning "LONG"   libraries/mathieeedp.h: *64
IEPointerPixel   structure tag size 0x0008 devices/inputevent.h: *88
IEPointerTablet structure tag (size 0x000a bytes)in struct IEPointerTablet
                 devices/inputevent.h: *108
IEQUALIFIERB_CAPSLOCK #define 2 = 0x00000002  devices/inputevent.h: *182
IEQUALIFIERB_CONTROL #define 3 = 0x00000003  devices/inputevent.h: *183
IEQUALIFIERB_INTERRUPT #define 10 = 0x0000000a  devices/inputevent.h: *190
IEQUALIFIERB_LALT #define 4 = 0x00000004  devices/inputevent.h: *184
IEQUALIFIERB_LCOMMAND #define 6 = 0x00000006  devices/inputevent.h: *186
IEQUALIFIERB_LEFTBUTTON #define 14 = 0x0000000e  devices/inputevent.h: *194
IEQUALIFIERB_LSHIFT #define 0 = 0x00000000  devices/inputevent.h: *180
IEQUALIFIERB_MIDBUTTON #define 12 = 0x0000000c  devices/inputevent.h: *192
IEQUALIFIERB_MULTIBROADCAST #define 11 = 0x0000000b
                 devices/inputevent.h: *191
IEQUALIFIERB_NUMERICPAD #define 8 = 0x00000008  devices/inputevent.h: *188
IEQUALIFIERB_RALT #define 5 = 0x00000005  devices/inputevent.h: *185
IEQUALIFIERB_RBUTTON #define 13 = 0x0000000d  devices/inputevent.h: *193
IEQUALIFIERB_RCOMMAND #define 7 = 0x00000007  devices/inputevent.h: *187
IEQUALIFIERB_RELATIVEMOUSE #define 15 = 0x0000000f  devices/inputevent.h: *195
IEQUALIFIERB_REPEAT #define 9 = 0x00000009  devices/inputevent.h: *189
IEQUALIFIERB_RSHIFT #define 1 = 0x00000001  devices/inputevent.h: *181
IEQUALIFIER_CAPSLOCK #define 0x0004 = 0x00000004  devices/inputevent.h: *165
IEQUALIFIER_CONTROL #define 0x0008 = 0x00000008  devices/inputevent.h: *166
IEQUALIFIER_INTERRUPT #define 0x0400 = 0x00000400  devices/inputevent.h: *173
IEQUALIFIER_LALT #define 0x0010 = 0x00000010  devices/inputevent.h: *167
IEQUALIFIER_LCOMMAND #define 0x0040 = 0x00000040  devices/inputevent.h: *169
IEQUALIFIER_LEFTBUTTON #define 0x4000 = 0x00004000  devices/inputevent.h: *177
```

```
IEQUALIFIER_LSHIFT #define 0x0001 = 0x00000001  devices/inputevent.h: *163
IEQUALIFIER_MIDBUTTON #define 0x1000 = 0x00001000  devices/inputevent.h: *175
IEQUALIFIER_MULTIBROADCAST #define 0x0800 = 0x00000800
                  devices/inputevent.h: *174
IEQUALIFIER_NUMERICPAD #define 0x0100 = 0x00000100  devices/inputevent.h: *171
IEQUALIFIER_RALT #define 0x0020 = 0x00000020  devices/inputevent.h: *168
IEQUALIFIER_RBUTTON #define 0x2000 = 0x00002000  devices/inputevent.h: *176
IEQUALIFIER_RCOMMAND #define 0x0080 = 0x00000080  devices/inputevent.h: *170
IEQUALIFIER_RELATIVEMOUSE #define 0x8000 = 0x00008000
                  devices/inputevent.h: *178
IEQUALIFIER_REPEAT #define 0x0200 = 0x00000200  devices/inputevent.h: *172
IEQUALIFIER_RSHIFT #define 0x0002 = 0x00000002  devices/inputevent.h: *164
IESUBCLASS_COMPATIBLE #define 0x00 = 0x00000000  devices/inputevent.h: *72
IESUBCLASS_PIXEL #define 0x01 = 0x00000001  devices/inputevent.h: *74
IESUBCLASS_TABLET #define 0x02 = 0x00000002  devices/inputevent.h: *76
IEvent            pointer to struct InputEvent in struct SGWork
         +0x0014 intuition/sghooks.h: *42
IFFCMD_CLEANUP    #define 1 = 0x00000001  libraries/iffparse.h: *189
IFFCMD_ENTRY      #define 5 = 0x00000005  libraries/iffparse.h: *193
IFFCMD_EXIT       #define 6 = 0x00000006  libraries/iffparse.h: *194
IFFCMD_INIT       #define 0 = 0x00000000  libraries/iffparse.h: *188
IFFCMD_PURGELCI   #define 7 = 0x00000007  libraries/iffparse.h: *195
IFFCMD_READ       #define 2 = 0x00000002  libraries/iffparse.h: *190
IFFCMD_SEEK       #define 4 = 0x00000004  libraries/iffparse.h: *192
IFFCMD_WRITE      #define 3 = 0x00000003  libraries/iffparse.h: *191
IFFERR_EOC        #define -2L = 0xfffffffe  libraries/iffparse.h: *130
IFFERR_EOF        #define -1L = 0xffffffff  libraries/iffparse.h: *129
IFFERR_MANGLED    #define -8L = 0xfffffff8  libraries/iffparse.h: *136
IFFERR_NOHOOK     #define -11L = 0xfffffff5  libraries/iffparse.h: *139
IFFERR_NOMEM      #define -4L = 0xfffffffc  libraries/iffparse.h: *132
IFFERR_NOSCOPE    #define -3L = 0xfffffffd  libraries/iffparse.h: *131
IFFERR_NOTIFF     #define -10L = 0xfffffff6  libraries/iffparse.h: *138
IFFERR_READ       #define -5L = 0xfffffffb  libraries/iffparse.h: *133
IFFERR_SEEK       #define -7L = 0xfffffff9  libraries/iffparse.h: *135
IFFERR_SYNTAX     #define -9L = 0xfffffff7  libraries/iffparse.h: *137
IFFERR_WRITE      #define -6L = 0xfffffffa  libraries/iffparse.h: *134
IFFF_FSEEK        #define (1L<<1) = 0x00000002  libraries/iffparse.h: *48
IFFF_READ         #define 0L = 0x00000000  libraries/iffparse.h: *45
IFFF_RESERVED     #define 0xFFFF0000L = 0xffff0000  libraries/iffparse.h: *50
IFFF_RSEEK        #define (1L<<2) = 0x00000004  libraries/iffparse.h: *49
IFFF_RWBITS       #define (IFFF_READ | IFFF_WRITE) = 0x00000001
                  libraries/iffparse.h: *47
IFFF_WRITE        #define 1L = 0x00000001  libraries/iffparse.h: *46
IFFHandle         structure tag size 0x000c libraries/iffparse.h: *35
IFFLCI_COLLECTION #define MAKE_ID('c','o','l','l') = 0x636f6c6c
                  libraries/iffparse.h: *158
IFFLCI_ENTRYHANDLER #define MAKE_ID('e','n','h','d') = 0x656e6864
                  libraries/iffparse.h: *159
IFFLCI_EXITHANDLER #define MAKE_ID('e','x','h','d') = 0x65786864
                  libraries/iffparse.h: *160
IFFLCI_PROP       #define MAKE_ID('p','r','o','p') = 0x70726f70
                  libraries/iffparse.h: *157
IFFPARSE_RAWSTEP #define 2L = 0x00000002  libraries/iffparse.h: *167
IFFPARSE_SCAN     #define 0L = 0x00000000  libraries/iffparse.h: *165
IFFPARSE_STEP     #define 1L = 0x00000001  libraries/iffparse.h: *166
IFFSCC_CLEANUP    #define IFFCMD_CLEANUP = 0x00000001
                  libraries/iffparse.h: *199
IFFSCC_INIT       #define IFFCMD_INIT = 0x00000000  libraries/iffparse.h: *198
IFFSCC_READ       #define IFFCMD_READ = 0x00000002  libraries/iffparse.h: *200
IFFSCC_SEEK       #define IFFCMD_SEEK = 0x00000004  libraries/iffparse.h: *202
IFFSCC_WRITE      #define IFFCMD_WRITE = 0x00000003  libraries/iffparse.h: *201
IFFSIZE_UNKNOWN   #define -1L = 0xffffffff  libraries/iffparse.h: *182
IFFSLI_PROP       #define 3L = 0x00000003  libraries/iffparse.h: *174
IFFSLI_ROOT       #define 1L = 0x00000001  libraries/iffparse.h: *172
IFFSLI_TOP        #define 2L = 0x00000002  libraries/iffparse.h: *173
IFFStreamCmd      structure tag size 0x000c libraries/iffparse.h: *56
```

```
IFF_IFFPARSE_H    #define  libraries/iffparse.h: *2
IFF_RETURN2CLIENT #define -12L = 0xfffffff4  libraries/iffparse.h: *140
IFont             pointer to struct TextFont in struct Window
         +0x0080 intuition/intuition.h: *895
IGNORE_DIMENSIONS #define 0x0000 = 0x00000000  intuition/preferences.h: *247
IMAGECLASS        #define "imageclass"  intuition/classusr.h: *43
IMAGE_ATTRIBUTES #define (IA_Dummy) = 0x80020000  intuition/iobsolete.h: *234
IMAGE_NEGATIVE    #define 0x01 = 0x00000001  intuition/preferences.h: *173
IMAGE_POSITIVE    #define 0x00 = 0x00000000  intuition/preferences.h: *172
IMPORT            #define extern  exec/types.h: *21
IM_BGPEN          macro (1 argument)   intuition/imageclass.h: *31
IM_BOX            macro (1 argument)   intuition/imageclass.h: *29
IM_DRAW           #define 0x202L = 0x00000202  intuition/imageclass.h: *115
IM_DRAWFRAME      #define 0x206L = 0x00000206  intuition/imageclass.h: *120
IM_ERASE          #define 0x204L = 0x00000204  intuition/imageclass.h: *117
IM_ERASEFRAME     #define 0x209L = 0x00000209  intuition/imageclass.h: *123
IM_FGPEN          macro (1 argument)   intuition/imageclass.h: *30
IM_FRAMEBOX       #define 0x207L = 0x00000207  intuition/imageclass.h: *121
IM_HITFRAME       #define 0x208L = 0x00000208  intuition/imageclass.h: *122
IM_HITTEST        #define 0x203L = 0x00000203  intuition/imageclass.h: *116
IM_ITEM           #define (NM_ITEM | MENU_IMAGE) = 0x00000082
                  libraries/gadtools.h: *140
IM_MOVE           #define 0x205L = 0x00000205  intuition/imageclass.h: *118
IM_SUB            #define (NM_SUB | MENU_IMAGE) = 0x00000083
                  libraries/gadtools.h: *141
INACTIVEWINDOW    #define IDCMP_INACTIVEWINDOW = 0x00080000
                  intuition/iobsolete.h: *133
INCLUDE_VERSION   #define 36 = 0x00000024  exec/types.h: *17
IND_ADDHANDLER    #define (CMD_NONSTD+0) = 0x00000009  devices/input.h: *19
IND_REMHANDLER    #define (CMD_NONSTD+1) = 0x0000000a  devices/input.h: *20
IND_SETMPORT      #define (CMD_NONSTD+5) = 0x0000000e  devices/input.h: *24
IND_SETMTRIG      #define (CMD_NONSTD+7) = 0x00000010  devices/input.h: *26
IND_SETMTYPE      #define (CMD_NONSTD+6) = 0x0000000f  devices/input.h: *25
IND_SETPERIOD     #define (CMD_NONSTD+4) = 0x0000000d  devices/input.h: *23
IND_SETTHRESH     #define (CMD_NONSTD+3) = 0x0000000c  devices/input.h: *22
IND_WRITEEVENT    #define (CMD_NONSTD+2) = 0x0000000b  devices/input.h: *21
INITBYTE          macro (2 arguments)   exec/initializers.h: *17
INITLONG          macro (2 arguments)   exec/initializers.h: *21
INITSTRUCT        macro (4 arguments)   exec/initializers.h: *25
INITWORD          macro (2 arguments)   exec/initializers.h: *18
INREQUEST         #define WFLG_INREQUEST = 0x00004000
                  intuition/iobsolete.h: *162
INST_DATA         macro (2 arguments)   intuition/classes.h: *48
INTB_AUD0         #define (7) = 0x00000007  hardware/intbits.h: *25
INTB_AUD1         #define (8) = 0x00000008  hardware/intbits.h: *24
INTB_AUD2         #define (9) = 0x00000009  hardware/intbits.h: *23
INTB_AUD3         #define (10) = 0x0000000a  hardware/intbits.h: *22
INTB_BLIT         #define (6) = 0x00000006  hardware/intbits.h: *26
INTB_COPER        #define (4) = 0x00000004  hardware/intbits.h: *28
INTB_DSKBLK       #define (1) = 0x00000001  hardware/intbits.h: *31
INTB_DSKSYNC      #define (12) = 0x0000000c  hardware/intbits.h: *20
INTB_EXTER        #define (13) = 0x0000000d  hardware/intbits.h: *19
INTB_INTEN        #define (14) = 0x0000000e  hardware/intbits.h: *18
INTB_NMI          #define 15 = 0x0000000f  exec/interrupts.h: *46
INTB_PORTS        #define (3) = 0x00000003  hardware/intbits.h: *29
INTB_RBF          #define (11) = 0x0000000b  hardware/intbits.h: *21
INTB_SETCLR       #define (15) = 0x0000000f  hardware/intbits.h: *15
INTB_SOFTINT      #define (2) = 0x00000002  hardware/intbits.h: *30
INTB_TBE          #define (0) = 0x00000000  hardware/intbits.h: *32
INTB_VERTB        #define (5) = 0x00000005  hardware/intbits.h: *27
INTEGERIDCMP      #define (IDCMP_GADGETUP) = 0x00000040
                  libraries/gadtools.h: *68
INTEGER_KIND      #define 3 = 0x00000003  libraries/gadtools.h: *37
INTEGER_SCALING   #define 0x0100 = 0x00000100  intuition/preferences.h: *253
INTERHEIGHT       #define 4 = 0x00000004  libraries/gadtools.h: *88
INTERLACE         #define 4 = 0x00000004  graphics/display.h: *24
```

Include File Cross Reference Page 75

```
INTERWIDTH        #define 8 = 0x00000008  libraries/gadtools.h: *87
INTF_AUD0         #define (1<<7) = 0x00000080  hardware/intbits.h: *44
INTF_AUD1         #define (1<<8) = 0x00000100  hardware/intbits.h: *43
INTF_AUD2         #define (1<<9) = 0x00000200  hardware/intbits.h: *42
INTF_AUD3         #define (1<<10) = 0x00000400  hardware/intbits.h: *41
INTF_BLIT         #define (1<<6) = 0x00000040  hardware/intbits.h: *45
INTF_COPER        #define (1<<4) = 0x00000010  hardware/intbits.h: *47
INTF_DSKBLK       #define (1<<1) = 0x00000002  hardware/intbits.h: *50
INTF_DSKSYNC      #define (1<<12) = 0x00001000  hardware/intbits.h: *39
INTF_EXTER        #define (1<<13) = 0x00002000  hardware/intbits.h: *38
INTF_INTEN        #define (1<<14) = 0x00004000  hardware/intbits.h: *37
INTF_NMI          #define (1<<15) = 0x00008000  exec/interrupts.h: *47
INTF_PORTS        #define (1<<3) = 0x00000008  hardware/intbits.h: *48
INTF_RBF          #define (1<<11) = 0x00000800  hardware/intbits.h: *40
INTF_SETCLR       #define (1<<15) = 0x00008000  hardware/intbits.h: *36
INTF_SOFTINT      #define (1<<2) = 0x00000004  hardware/intbits.h: *49
INTF_TBE          #define (1<<0) = 0x00000001  hardware/intbits.h: *51
INTF_VERTB        #define (1<<5) = 0x00000020  hardware/intbits.h: *46
INTUITICKS        #define IDCMP_INTUITICKS = 0x00400000
                  intuition/iobsolete.h: *136
INTUITION_CGHOOKS_H #define 1 = 0x00000001  intuition/cghooks.h: *2
INTUITION_CLASSES_H #define 1 = 0x00000001  intuition/classes.h: *2
INTUITION_CLASSUSR_H #define 1 = 0x00000001  intuition/classusr.h: *2, 1
INTUITION_GADGETCLASS_H #define 1 = 0x00000001  intuition/gadgetclass.h: *2
INTUITION_ICCLASS_H #define  intuition/icclass.h: *2
INTUITION_IMAGECLASS_H #define TRUE = 0x00000001  intuition/imageclass.h: *2
INTUITION_INTUITIONBASE_H #define 1 = 0x00000001
                  intuition/intuitionbase.h: *2
INTUITION_INTUITION_H #define TRUE = 0x00000001  intuition/intuition.h: *2, 1
                  intuition/iobsolete.h: 38
                  intuition/cghooks.h: 19
                  intuition/gadgetclass.h: 19
                  intuition/intuitionbase.h: 23
                  libraries/gadtools.h: 25
                  workbench/workbench.h: 31
INTUITION_IOBSOLETE_H #define  intuition/iobsolete.h: *2, 1
                  intuition/intuition.h: 1357
                  intuition/gadgetclass.h: 244
                  intuition/imageclass.h: 203
INTUITION_PREFERENCES_H #define TRUE = 0x00000001
                  intuition/preferences.h: *2, 1
INTUITION_SCREENS_H #define TRUE = 0x00000001  intuition/screens.h: *2, 1
INTUITION_SGHOOKS_H #define TRUE = 0x00000001  intuition/sghooks.h: *2
INVALID_ID        #define ~0 = 0xffffffff  graphics/displayinfo.h: *141
INVERSVID         #define 4 = 0x00000004  graphics/rastport.h: *97
IOAudio           structure tag size 0x0044 devices/audio.h: *47
IOB_QUICK         #define 0 = 0x00000000  exec/io.h: *47
IOClipReq         structure tag size 0x0034 devices/clipboard.h: *43
                  libraries/iffparse.h: 117
IODRPReq          structure tag size 0x003e devices/printer.h: *156
IOERR_ABORTED     #define (-2) = 0xfffffffe  exec/errors.h: *16
IOERR_BADADDRESS  #define (-5) = 0xfffffffb  exec/errors.h: *19
IOERR_BADLENGTH   #define (-4) = 0xfffffffc  exec/errors.h: *18
IOERR_NOCMD       #define (-3) = 0xfffffffd  exec/errors.h: *17
IOERR_OPENFAIL    #define (-1) = 0xffffffff  exec/errors.h: *15
IOERR_SELFTEST    #define (-7) = 0xfffffff9  exec/errors.h: *21
IOERR_UNITBUSY    #define (-6) = 0xfffffffa  exec/errors.h: *20
IOExtPar          structure tag size 0x003e devices/parallel.h: *29
                  devices/prtbase.h: 76, 84
IOExtSer          structure tag size 0x0052 devices/serial.h: *38
                  devices/prtbase.h: 77, 85
IOExtTD           structure tag size 0x0038 devices/trackdisk.h: *119
IOF_QUICK         #define (1<<0) = 0x00000001  exec/io.h: *48
IOPARB_ABORT      #define 5 = 0x00000005  devices/parallel.h: *75
IOPARB_ACTIVE     #define 4 = 0x00000004  devices/parallel.h: *77
IOPARB_QUEUED     #define 6 = 0x00000006  devices/parallel.h: *73
```

Include File Cross Reference Page 76

```
IOPARF_ABORT      #define (1<<5) = 0x00000020  devices/parallel.h: *76
IOPARF_ACTIVE     #define (1<<4) = 0x00000010  devices/parallel.h: *78
IOPARF_QUEUED     #define (1<<6) = 0x00000040  devices/parallel.h: *74
IOPArray          structure tag size 0x0008 devices/parallel.h: *19, 55
IOPTB_PAPEROUT    #define 1 = 0x00000001  devices/parallel.h: *85
IOPTB_PARBUSY     #define 0 = 0x00000000  devices/parallel.h: *87
IOPTB_PARSEL      #define 2 = 0x00000002  devices/parallel.h: *81
IOPTB_RWDIR       #define 3 = 0x00000003  devices/parallel.h: *79
IOPTF_PAPEROUT    #define (1<<1) = 0x00000002  devices/parallel.h: *86
IOPTF_PARBUSY     #define (1<<0) = 0x00000001  devices/parallel.h: *88
IOPTF_PARSEL      #define (1<<2) = 0x00000004  devices/parallel.h: *84
IOPTF_RWDIR       #define (1<<3) = 0x00000008  devices/parallel.h: *80
IOPar             struct IOStdReq(size 0x0030 bytes) in struct IOExtPar
          +0x0000 devices/parallel.h: *30
IOPrtCmdReq       structure tag size 0x0026 devices/printer.h: *142
IORequest         structure tag size 0x0020 exec/io.h: *20
                  devices/audio.h: 48
                  devices/timer.h: 38
IOSer             struct IOStdReq(size 0x0030 bytes) in struct IOExtSer
          +0x0000 devices/serial.h: *39
IOStdReq          structure tag size 0x0030 exec/io.h: *29
                  devices/narrator.h: 94
                  devices/parallel.h: 30
                  devices/serial.h: 39
                  devices/trackdisk.h: 120
IOTArray          structure tag size 0x0008 devices/serial.h: *22, 67
IOTDB_INDEXSYNC   #define 4 = 0x00000004  devices/trackdisk.h: *173
IOTDB_WORDSYNC    #define 5 = 0x00000005  devices/trackdisk.h: *179
IOTDF_INDEXSYNC   #define (1<<4) = 0x00000010  devices/trackdisk.h: *174
IOTDF_WORDSYNC    #define (1<<5) = 0x00000020  devices/trackdisk.h: *180
IO_STATB_OVERRUN #define 8 = 0x00000008  devices/serial.h: *126
IO_STATB_READBREAK #define 10 = 0x0000000a  devices/serial.h: *122
IO_STATB_WROTEBREAK #define 9 = 0x00000009  devices/serial.h: *124
IO_STATB_XOFFREAD #define 12 = 0x0000000c  devices/serial.h: *118
IO_STATB_XOFFWRITE #define 11 = 0x0000000b  devices/serial.h: *120
IO_STATF_OVERRUN #define (1<<8) = 0x00000100  devices/serial.h: *127
IO_STATF_READBREAK #define (1<<10) = 0x00000400  devices/serial.h: *123
IO_STATF_WROTEBREAK #define (1<<9) = 0x00000200  devices/serial.h: *125
IO_STATF_XOFFREAD #define (1<<12) = 0x00001000  devices/serial.h: *119
IO_STATF_XOFFWRITE #define (1<<11) = 0x00000800  devices/serial.h: *121
ISDRAWN           #define 0x1000 = 0x00001000  intuition/intuition.h: *135
ISGRTRX           #define 4 = 0x00000004  graphics/clip.h: *85
ISGRTRY           #define 8 = 0x00000008  graphics/clip.h: *86
ISLESSX           #define 1 = 0x00000001  graphics/clip.h: *83
ISLESSY           #define 2 = 0x00000002  graphics/clip.h: *84
ITEMENABLED       #define 0x0010 = 0x00000010  intuition/intuition.h: *122
ITEMNUM           macro (1 argument)  intuition/intuition.h: *1267
ITEMTEXT          #define 0x0002 = 0x00000002  intuition/intuition.h: *119
ITEM_EQUAL        #define -2 = 0xfffffffe  dos/dos.h: *230
ITEM_ERROR        #define -1 = 0xffffffff  dos/dos.h: *231
ITEM_NOTHING      #define 0 = 0x00000000  dos/dos.h: *232
ITEM_QUOTED       #define 2 = 0x00000002  dos/dos.h: *234
ITEM_UNQUOTED     #define 1 = 0x00000001  dos/dos.h: *233
IText             pointer to unsigned char in struct IntuiText
          +0x000c intuition/intuition.h: *577
ITextFont         pointer to struct TextAttr in struct IntuiText
          +0x0008 intuition/intuition.h: *576
IVALUE            macro (1 argument)   rexx/storage.h: *51
IX                "LONG" libraries/commodities.h: *192
IXSYM_ALT         #define 4 = 0x00000004  libraries/commodities.h: *197
IXSYM_ALTMASK     #define (IEQUALIFIER_LALT | IEQUALIFIER_RALT) = 0x00000030
                  libraries/commodities.h: *202
IXSYM_CAPS        #define 2 = 0x00000002  libraries/commodities.h: *196
IXSYM_CAPSMASK    #define (IXSYM_SHIFTMASK | IEQUALIFIER_CAPSLOCK) = 0x00000007
                  libraries/commodities.h: *201
IXSYM_SHIFT       #define 1 = 0x00000001  libraries/commodities.h: *195
```

```
IXSYM_SHIFTMASK   #define (IEQUALIFIER_LSHIFT | IEQUALIFIER_RSHIFT) = 0x00000003
                  libraries/commodities.h: *200
IX_NORMALQUALS    #define 0x7FFF;  libraries/commodities.h: *204
IX_VERSION        #define 2 = 0x00000002  libraries/commodities.h: *171
IdleCount         unsigned long int in struct ExecBase
         +0x0118  exec/execbase.h: *65
Image             structure tag
     size 0x0014  intuition/intuition.h: *174, 619, 666, 865, 996,
                          1054
ImageBMap         pointer to struct BitMap in struct Requester
         +0x0044  intuition/intuition.h: *171
ImageData         pointer to unsigned short int in struct Image
         +0x000a  intuition/intuition.h: *626
ImageData         pointer to short int in struct VSprite
         +0x0024  graphics/gels.h: *110
ImageShadow       pointer to short int in struct Bob
         +0x0006  graphics/gels.h: *153
InfoData          structure tag size 0x0024 dos/dos.h: *119
InitAnimate       macro (1 argument)   graphics/gels.h: *252
InitialModes      unsigned long int in struct StringExtend
         +0x0008  intuition/sghooks.h: *26
InputEvent        structure tag size 0x0016 devices/inputevent.h: *199, 200
                  intuition/gadgetclass.h: 204
                  intuition/sghooks.h: 42
InputXpression    structure tag size 0x000c libraries/commodities.h: *173, 192
IntVector         structure tag size 0x000c exec/interrupts.h: *31
                  exec/execbase.h: 59
IntVects          array [16] of struct IntVector(size 0x000c bytes) in struct
                  ExecBase
         +0x0054  exec/execbase.h: *59
Interrupt         structure tag size 0x0016 exec/interrupts.h: *24
                  graphics/gfxbase.h: 36
                  resources/disk.h: 45, 46, 47, 59, 60, 61
IntrList          struct List(size 0x000e bytes) in struct ExecBase
         +0x016c  exec/execbase.h: *89
IntuiMessage      structure tag
     size 0x0034  intuition/intuition.h: *677, 716, 857
IntuiText         structure tag
     size 0x0014  intuition/intuition.h: *155, 240, 570, 578
IntuitionBase     structure tag size 0x0050 intuition/intuitionbase.h: *68
IoBuff            structure tag size 0x0100 rexx/rexxio.h: *25
Iptr              pointer to struct Isrvstr in struct Isrvstr
         +0x000e  graphics/graphint.h: *23
IsListEmpty       macro (1 argument)   exec/lists.h: *53
IsMsgPortEmpty    macro (1 argument)   exec/lists.h: *56
Isrvstr           structure tag size 0x001e graphics/graphint.h: *20, 23
ItemFill          pointer to void in struct MenuItem
         +0x0012  intuition/intuition.h: *99
joy0dat           unsigned short int in struct Custom
         +0x000a  hardware/custom.h: *33
joy1dat           unsigned short int in struct Custom
         +0x000c  hardware/custom.h: *34
joytest           unsigned short int in struct Custom
         +0x0036  hardware/custom.h: *54
jump              array [4] of unsigned short int in struct copinit
         +0x0064  graphics/copper.h: *101
JAM1              #define 0 = 0x00000000  graphics/rastport.h: *94
JAM2              #define 1 = 0x00000001  graphics/rastport.h: *95
JazzX             short int in struct Menu +0x0016 intuition/intuition.h: *72
JazzY             short int in struct Menu +0x0018 intuition/intuition.h: *72
km_HiCapsable     pointer to unsigned char in struct KeyMap
         +0x0018  devices/keymap.h: *30
km_HiKeyMap       pointer to unsigned long int in struct KeyMap
         +0x0014  devices/keymap.h: *29
km_HiKeyMapTypes  pointer to unsigned char in struct KeyMap
         +0x0010  devices/keymap.h: *28
```

```
km_HiRepeatable pointer to unsigned char in struct KeyMap
        +0x001c devices/keymap.h: *31
km_LoCapsable    pointer to unsigned char in struct KeyMap
        +0x0008 devices/keymap.h: *26
km_LoKeyMap      pointer to unsigned long int in struct KeyMap
        +0x0004 devices/keymap.h: *25
km_LoKeyMapTypes pointer to unsigned char in struct KeyMap
        +0x0000 devices/keymap.h: *24
km_LoRepeatable pointer to unsigned char in struct KeyMap
        +0x000c devices/keymap.h: *27
kn_KeyMap        struct KeyMap(size 0x0020 bytes) in struct KeyMapNode
        +0x000e devices/keymap.h: *36
kn_Node          struct Node(size 0x000e bytes) in struct KeyMapNode
        +0x0000 devices/keymap.h: *35
kr_List          struct List(size 0x000e bytes) in struct KeyMapResource
        +0x000e devices/keymap.h: *42
kr_Node          struct Node(size 0x000e bytes) in struct KeyMapResource
        +0x0000 devices/keymap.h: *41
KBD_ADDRESETHANDLER #define (CMD_NONSTD+2) = 0x0000000b
                 devices/keyboard.h: *21
KBD_READEVENT    #define (CMD_NONSTD+0) = 0x00000009  devices/keyboard.h: *19
KBD_READMATRIX   #define (CMD_NONSTD+1) = 0x0000000a  devices/keyboard.h: *20
KBD_REMRESETHANDLER #define (CMD_NONSTD+3) = 0x0000000c
                 devices/keyboard.h: *22
KBD_RESETHANDLERDONE #define (CMD_NONSTD+4) = 0x0000000d
                 devices/keyboard.h: *23
KCB_ALT          #define 1 = 0x00000001  devices/keymap.h: *50
KCB_CONTROL      #define 2 = 0x00000002  devices/keymap.h: *52
KCB_DEAD         #define 5 = 0x00000005  devices/keymap.h: *57
KCB_DOWNUP       #define 3 = 0x00000003  devices/keymap.h: *54
KCB_NOP          #define 7 = 0x00000007  devices/keymap.h: *63
KCB_SHIFT        #define 0 = 0x00000000  devices/keymap.h: *48
KCB_STRING       #define 6 = 0x00000006  devices/keymap.h: *60
KCF_ALT          #define 0x02 = 0x00000002  devices/keymap.h: *51
KCF_CONTROL      #define 0x04 = 0x00000004  devices/keymap.h: *53
KCF_DEAD         #define 0x20 = 0x00000020  devices/keymap.h: *58
KCF_DOWNUP       #define 0x08 = 0x00000008  devices/keymap.h: *55
KCF_NOP          #define 0x80 = 0x00000080  devices/keymap.h: *64
KCF_SHIFT        #define 0x01 = 0x00000001  devices/keymap.h: *49
KCF_STRING       #define 0x40 = 0x00000040  devices/keymap.h: *61
KC_NOQUAL        #define 0 = 0x00000000  devices/keymap.h: *46
KC_VANILLA       #define 7 = 0x00000007  devices/keymap.h: *47
KEEPNUM          #define (NSF_STRING | NSF_SOURCE | NSF_NUMBER | NSF_BINARY) =
                 0x0000009a
                 rexx/storage.h: *79
KEEPSTR          #define (NSF_STRING | NSF_SOURCE | NSF_NOTNUM) = 0x00000086
                 rexx/storage.h: *78
KEYCODE_B        #define 0x35 = 0x00000035  intuition/intuition.h: *1350
KEYCODE_GREATER  #define 0x39 = 0x00000039  intuition/intuition.h: *1354
KEYCODE_LESS     #define 0x38 = 0x00000038  intuition/intuition.h: *1353
KEYCODE_M        #define 0x37 = 0x00000037  intuition/intuition.h: *1352
KEYCODE_N        #define 0x36 = 0x00000036  intuition/intuition.h: *1351
KEYCODE_Q        #define 0x10 = 0x00000010  intuition/intuition.h: *1346
KEYCODE_V        #define 0x34 = 0x00000034  intuition/intuition.h: *1349
KEYCODE_X        #define 0x32 = 0x00000032  intuition/intuition.h: *1348
KEYCODE_Z        #define 0x31 = 0x00000031  intuition/intuition.h: *1347
KNOBHIT          #define 0x0100 = 0x00000100  intuition/intuition.h: *507
KNOBHMIN         #define 6 = 0x00000006  intuition/intuition.h: *512
KNOBVMIN         #define 4 = 0x00000004  intuition/intuition.h: *513
KeyMap           structure tag size 0x0020 devices/keymap.h: *23, 36
                 devices/conunit.h: 76
                 intuition/intuition.h: 560
KeyMapNode       structure tag size 0x002e devices/keymap.h: *34
KeyMapResource   structure tag size 0x001c devices/keymap.h: *40
KeyRptDelay      struct timeval(size 0x0008 bytes) in struct Preferences
        +0x000c intuition/preferences.h: *57
```

Include File Cross Reference Page 79

```
KeyRptSpeed      struct timeval(size 0x0008 bytes) in struct Preferences
         +0x0004 intuition/preferences.h: *56
KickCheckSum     pointer to void in struct ExecBase
         +0x022a exec/execbase.h: *124
KickMemPtr       pointer to void in struct ExecBase
         +0x0222 exec/execbase.h: *122
KickTagPtr       pointer to void in struct ExecBase
         +0x0226 exec/execbase.h: *123
l_pad            unsigned char in struct List +0x000d exec/lists.h: *27
lastBlissObj     pointer to void in struct GelsInfo
         +0x0022 graphics/rastport.h: *53
lastColor        pointer to pointer to short int in struct GelsInfo
         +0x000e graphics/rastport.h: *50
lci_ID           unsigned long int in struct LocalContextItem
         +0x0008 libraries/iffparse.h: *85
lci_Ident        unsigned long int in struct LocalContextItem
         +0x0010 libraries/iffparse.h: *87
lci_Node         struct MinNode(size 0x0008 bytes) in struct LocalContextItem
         +0x0000 libraries/iffparse.h: *84
lci_Type         unsigned long int in struct LocalContextItem
         +0x000c libraries/iffparse.h: *86
leftmost         short int in struct GelsInfo +0x0016 graphics/rastport.h: *52
lh_Flags         #define lib_Flags  exec/libraries.h: *57
lh_Head          pointer to struct Node in struct List
         +0x0000 exec/lists.h: *23
lh_IdString      #define lib_IdString  exec/libraries.h: *63
lh_NegSize       #define lib_NegSize  exec/libraries.h: *59
lh_Node          #define lib_Node  exec/libraries.h: *56
lh_OpenCnt       #define lib_OpenCnt  exec/libraries.h: *65
lh_PosSize       #define lib_PosSize  exec/libraries.h: *60
lh_Revision      #define lib_Revision  exec/libraries.h: *62
lh_Sum           #define lib_Sum  exec/libraries.h: *64
lh_Tail          pointer to struct Node in struct List
         +0x0004 exec/lists.h: *24
lh_TailPred      pointer to struct Node in struct List
         +0x0008 exec/lists.h: *25
lh_Type          unsigned char in struct List +0x000c exec/lists.h: *26
lh_Version       #define lib_Version  exec/libraries.h: *61
lh_pad           #define lib_pad  exec/libraries.h: *58
lib_Flags        unsigned char in struct Library +0x000e exec/libraries.h: *37
lib_IdString     pointer to void in struct Library
         +0x0018 exec/libraries.h: *43
lib_NegSize      unsigned short int in struct Library
         +0x0010 exec/libraries.h: *39
lib_Node         struct Node(size 0x000e bytes) in struct Library
         +0x0000 exec/libraries.h: *36
lib_OpenCnt      unsigned short int in struct Library
         +0x0020 exec/libraries.h: *45
lib_PosSize      unsigned short int in struct Library
         +0x0012 exec/libraries.h: *40
lib_Revision     unsigned short int in struct Library
         +0x0016 exec/libraries.h: *42
lib_Sum          unsigned long int in struct Library
         +0x001c exec/libraries.h: *44
lib_Version      unsigned short int in struct Library
         +0x0014 exec/libraries.h: *41
lib_pad          unsigned char in struct Library +0x000f exec/libraries.h: *38
linpatcnt        char in struct RastPort +0x001e graphics/rastport.h: *70
ln_Name          pointer to char in struct Node +0x000a exec/nodes.h: *29
ln_Pred          pointer to struct Node in struct Node
         +0x0004 exec/nodes.h: *26
ln_Pri           char in struct Node +0x0009 exec/nodes.h: *28
ln_Succ          pointer to struct Node in struct Node
         +0x0000 exec/nodes.h: *25
ln_Type          unsigned char in struct Node +0x0008 exec/nodes.h: *27
lobs             pointer to struct Layer in struct ClipRect
```

Include File Cross Reference Page 80

```
         +0x0008 graphics/clip.h: *68
log              #define IEEEDPLog  libraries/mathffp.h: *47
                 libraries/mathieeedp.h: *47
log10            #define IEEEDPLog10  libraries/mathffp.h: *48
                 libraries/mathieeedp.h: *48
longreserved     array [2] of unsigned long int in struct RastPort
         +0x0046 graphics/rastport.h: *86
longreserved     long int in struct Layer_Info +0x0054 graphics/layers.h: *42
lsb_ChkSum       long int in struct LoadSegBlock
         +0x0008 devices/hardblocks.h: *192
lsb_HostID       unsigned long int in struct LoadSegBlock
         +0x000c devices/hardblocks.h: *193
lsb_ID           unsigned long int in struct LoadSegBlock
         +0x0000 devices/hardblocks.h: *190
lsb_LoadData     array [123] of unsigned long int in struct LoadSegBlock
         +0x0014 devices/hardblocks.h: *195
lsb_Next         unsigned long int in struct LoadSegBlock
         +0x0010 devices/hardblocks.h: *194
lsb_SummedLongs unsigned long int in struct LoadSegBlock
         +0x0004 devices/hardblocks.h: *191
lv_Flags         unsigned short int in struct LocalVar +0x000e dos/var.h: *29
lv_Len           unsigned long int in struct LocalVar +0x0014 dos/var.h: *31
lv_Node          struct Node(size 0x000e bytes) in struct LocalVar
         +0x0000 dos/var.h: *28
lv_Value         pointer to unsigned char in struct LocalVar
         +0x0010 dos/var.h: *30
LABELIMAGE       #define GFLG_LABELIMAGE = 0x00002000
                 intuition/iobsolete.h: *64
LABELITEXT       #define GFLG_LABELITEXT = 0x00000000
                 intuition/iobsolete.h: *62
LABELMASK        #define GFLG_LABELMASK = 0x00003000
                 intuition/iobsolete.h: *61
LABELSTRING      #define GFLG_LABELSTRING = 0x00001000
                 intuition/iobsolete.h: *63
LACE             #define 0x0004 = 0x00000004  graphics/view.h: *91
LACEWB           #define (1<< 0) = 0x00000001  intuition/preferences.h: *133
LAYERBACKDROP    #define 0x40 = 0x00000040  graphics/layers.h: *27
LAYERREFRESH     #define 0x80 = 0x00000080  graphics/layers.h: *28
LAYERSIMPLE      #define 1 = 0x00000001  graphics/layers.h: *23
LAYERSMART       #define 2 = 0x00000002  graphics/layers.h: *24
LAYERSUPER       #define 4 = 0x00000004  graphics/layers.h: *25
LAYERUPDATING    #define 0x10 = 0x00000010  graphics/layers.h: *26
LAYER_CLIPRECTS_LOST #define 0x100 = 0x00000100  graphics/layers.h: *29
LAYOUTA_Dummy    #define (TAG_USER + 0x38000) = 0x80038000
                 intuition/gadgetclass.h: *151
LAYOUTA_LAYOUTOBJ #define LAYOUTA_LayoutObj = 0x80038001
                 intuition/iobsolete.h: *225
LAYOUTA_LayoutObj #define (LAYOUTA_Dummy + 0x0001) = 0x80038001
                 intuition/gadgetclass.h: *152
LAYOUTA_ORIENTATION #define LAYOUTA_Orientation = 0x80038003
                 intuition/iobsolete.h: *227
LAYOUTA_Orientation #define (LAYOUTA_Dummy + 0x0003) = 0x80038003
                 intuition/gadgetclass.h: *154
LAYOUTA_SPACING  #define LAYOUTA_Spacing = 0x80038002
                 intuition/iobsolete.h: *226
LAYOUTA_Spacing  #define (LAYOUTA_Dummy + 0x0002) = 0x80038002
                 intuition/gadgetclass.h: *153
LCMptr           pointer to unsigned short int in struct GfxBase
         +0x00e4 graphics/gfxbase.h: *64
LDB_ASSIGNS      #define 4 = 0x00000004  dos/dosextens.h: *428
LDB_DEVICES      #define 2 = 0x00000002  dos/dosextens.h: *424
LDB_ENTRY        #define 5 = 0x00000005  dos/dosextens.h: *430
LDB_READ         #define 0 = 0x00000000  dos/dosextens.h: *434
LDB_VOLUMES      #define 3 = 0x00000003  dos/dosextens.h: *426
LDB_WRITE        #define 1 = 0x00000001  dos/dosextens.h: *436
LDF_ALL          #define (LDF_DEVICES|LDF_VOLUMES|LDF_ASSIGNS) = 0x0000001c
```

```
                 dos/dosextens.h: *440
LDF_ASSIGNS      #define (1L << LDB_ASSIGNS) = 0x00000010
                 dos/dosextens.h: *429
LDF_DEVICES      #define (1L << LDB_DEVICES) = 0x00000004
                 dos/dosextens.h: *425
LDF_ENTRY        #define (1L << LDB_ENTRY) = 0x00000020  dos/dosextens.h: *431
LDF_READ         #define (1L << LDB_READ) = 0x00000001  dos/dosextens.h: *435
LDF_VOLUMES      #define (1L << LDB_VOLUMES) = 0x00000008
                 dos/dosextens.h: *427
LDF_WRITE        #define (1L << LDB_WRITE) = 0x00000002  dos/dosextens.h: *437
LEFTBORDER       #define GACT_LEFTBORDER = 0x00000020
                 intuition/iobsolete.h: *74
LEFTHIT          #define 4 = 0x00000004  graphics/collide.h: *34
LEFTIMAGE        #define (0x0AL) = 0x0000000a  intuition/imageclass.h: *106
LEN_DATSTRING    #define 16 = 0x00000010  dos/datetime.h: *37
LETTER           #define 0x100 = 0x00000100  intuition/preferences.h: *165
LIBF_CHANGED     #define (1<<1) = 0x00000002  exec/libraries.h: *50
LIBF_DELEXP      #define (1<<3) = 0x00000008  exec/libraries.h: *52
LIBF_SUMMING     #define (1<<0) = 0x00000001  exec/libraries.h: *49
LIBF_SUMUSED     #define (1<<2) = 0x00000004  exec/libraries.h: *51
LIBRARIES_ASL_H  #define 1 = 0x00000001  libraries/asl.h: *2
LIBRARIES_COMMODITIES_H #define  libraries/commodities.h: *2
LIBRARIES_CONFIGREGS_H #define  libraries/configregs.h: *2
                 libraries/configvars.h: 23
LIBRARIES_CONFIGVARS_H #define  libraries/configvars.h: *2
                 libraries/expansionbase.h: 27
LIBRARIES_DISKFONT_H #define  libraries/diskfont.h: *2
LIBRARIES_DOSEXTENS_H #define  libraries/dosextens.h: *2, 1
LIBRARIES_DOS_H  #define  libraries/dos.h: *2, 1
LIBRARIES_EXPANSIONBASE_H #define  libraries/expansionbase.h: *2
LIBRARIES_EXPANSION_H #define  libraries/expansion.h: *2
LIBRARIES_FILEHANDLER_H #define  libraries/filehandler.h: *2
LIBRARIES_GADTOOLS_H #define  libraries/gadtools.h: *2
LIBRARIES_MATHFFP_H #define 1 = 0x00000001  libraries/mathffp.h: *2
LIBRARIES_MATHIEEEDP_H #define  libraries/mathieeedp.h: *2
LIBRARIES_MATHLIBRARY_H #define  libraries/mathlibrary.h: *2
LIBRARIES_TRANSLATOR_H #define  libraries/translator.h: *2
LIBRARY_MINIMUM  #define 33 = 0x00000021  exec/types.h: *85
LIB_BASE         #define (-LIB_VECTSIZE) = 0xfffffffa  exec/libraries.h: *23
LIB_CLOSE        #define (-12) = 0xfffffff4  exec/libraries.h: *29
LIB_EXPUNGE      #define (-18) = 0xffffffee  exec/libraries.h: *30
LIB_EXTFUNC      #define (-24) = 0xffffffe8  exec/libraries.h: *31
LIB_NONSTD       #define (LIB_USERDEF) = 0xffffffe2  exec/libraries.h: *25
LIB_OPEN         #define (-6) = 0xfffffffa  exec/libraries.h: *28
LIB_RESERVED     #define 4 = 0x00000004  exec/libraries.h: *22
LIB_USERDEF      #define (LIB_BASE-(LIB_RESERVED*LIB_VECTSIZE)) = 0xffffffe2
                 exec/libraries.h: *24
LIB_VECTSIZE     #define 6 = 0x00000006  exec/libraries.h: *21
LINEMODE         #define 0x1 = 0x00000001  hardware/blit.h: *67
LINK_HARD        #define 0 = 0x00000000  dos/dos.h: *226
LINK_SOFT        #define 1 = 0x00000001  dos/dos.h: *227
LISTVIEWIDCMP    #define (IDCMP_GADGETUP | IDCMP_GADGETDOWN | IDCMP_MOUSEMOVE
                 | ARROWIDCMP) = 0x00400078
                 libraries/gadtools.h: *70
LISTVIEW_KIND    #define 4 = 0x00000004  libraries/gadtools.h: *38
LLOFFSET         macro (1 argument)   rexx/rexxio.h: *51
LLVERS           macro (1 argument)   rexx/rexxio.h: *52
LMN_REGION       #define -1 = 0xffffffff  graphics/layers.h: *32
LOCK_DIFFERENT   #define -1 = 0xffffffff  dos/dos.h: *219
LOCK_SAME        #define 0 = 0x00000000  dos/dos.h: *217
LOCK_SAME_HANDLER #define 1 = 0x00000001  dos/dos.h: *218
LOFCprList       pointer to struct cprlist in struct View
         +0x0004 graphics/view.h: *61
LOFlist          pointer to unsigned short int in struct GfxBase
         +0x0032 graphics/gfxbase.h: *32
LOG10            #define ((double) 2.302585092994046)
```

```
                 libraries/mathffp.h: *24
                 libraries/mathieeedp.h: *26
LOLDIS           #define 0x0800 = 0x00000800  hardware/custom.h: *148
LONELYMESSAGE    #define IDCMP_LONELYMESSAGE = 0x80000000
                 intuition/iobsolete.h: *140
LONG             long int  exec/types.h: *38
LONGBITS         unsigned long int  exec/types.h: *40
LONGINT          #define GACT_LONGINT = 0x00000800  intuition/iobsolete.h: *83
LORESDPF2_KEY    #define 0x00000440 = 0x00000440  graphics/displayinfo.h: *169
LORESDPF_KEY     #define 0x00000400 = 0x00000400  graphics/displayinfo.h: *163
LORESLACEDPF2_KEY #define 0x00000444 = 0x00000444
                 graphics/displayinfo.h: *172
LORESLACEDPF_KEY #define 0x00000404 = 0x00000404  graphics/displayinfo.h: *166
LORESLACE_KEY    #define 0x00000004 = 0x00000004  graphics/displayinfo.h: *159
LORES_KEY        #define 0x00000000 = 0x00000000  graphics/displayinfo.h: *155
LORIENT_HORIZ    #define 1 = 0x00000001  intuition/gadgetclass.h: *158
LORIENT_NONE     #define 0 = 0x00000000  intuition/gadgetclass.h: *157
LORIENT_VERT     #define 2 = 0x00000002  intuition/gadgetclass.h: *159
LOWCHECKWIDTH    #define 13 = 0x0000000d  intuition/intuition.h: *1302
LOWCOMMWIDTH     #define 16 = 0x00000010  intuition/intuition.h: *1303
LOWRESGADGET     #define 1 = 0x00000001  intuition/intuitionbase.h: *44
LOWRESPICK       #define 0x0001 = 0x00000001  intuition/intuitionbase.h: *37
LVB_IGNORE       #define 7 = 0x00000007  dos/var.h: *43
LVF_IGNORE       #define 0x80 = 0x00000080  dos/var.h: *44
LV_ALIAS         #define 1 = 0x00000001  dos/var.h: *41
LV_VAR           #define 0 = 0x00000000  dos/var.h: *40
LW_RESERVED      #define 1 = 0x00000001  intuition/preferences.h: *134
LaceWB           unsigned char in struct Preferences
         +0x00b9 intuition/preferences.h: *110
LastAlert        array [4] of long int in struct ExecBase
         +0x0202 exec/execbase.h: *99
LastChanceMemory pointer to struct SignalSemaphore in struct GfxBase
         +0x00e0 graphics/gfxbase.h: *63
Layer            structure tag size 0x00a0 graphics/clip.h: *34, 36, 68
                 graphics/rastport.h: 58
                 graphics/layers.h: 36, 37
                 intuition/intuition.h: 161, 890
                 intuition/screens.h: 145
                 intuition/cghooks.h: 38
Layer            pointer to struct Layer in struct RastPort
         +0x0000 graphics/rastport.h: *58
LayerInfo        pointer to struct Layer_Info in struct Layer
         +0x0044 graphics/clip.h: *51
LayerInfo        struct Layer_Info(size 0x0066 bytes) in struct Screen
         +0x00e0 intuition/screens.h: *130
LayerInfo_extra pointer to void in struct Layer_Info
         +0x0062 graphics/layers.h: *48
LayerInfo_extra_size unsigned short int in struct Layer_Info
         +0x005c graphics/layers.h: *46
Layer_Info       structure tag size 0x0066 graphics/clip.h: *51
                 graphics/layers.h: 34
                 intuition/screens.h: 130
Left             short int in struct IBox +0x0000 intuition/intuition.h: *784
LeftBorder       unsigned short int in struct PropInfo
         +0x0012 intuition/intuition.h: *491
LeftEdge         short int in struct Menu +0x0004 intuition/intuition.h: *65
LeftEdge         short int in struct MenuItem
         +0x0004 intuition/intuition.h: *93
LeftEdge         short int in struct Requester
         +0x0004 intuition/intuition.h: *149
LeftEdge         short int in struct Gadget +0x0004 intuition/intuition.h: *220
LeftEdge         short int in struct IntuiText
         +0x0004 intuition/intuition.h: *574
LeftEdge         short int in struct Border +0x0000 intuition/intuition.h: *600
LeftEdge         short int in struct Image +0x0000 intuition/intuition.h: *621
LeftEdge         short int in struct Window +0x0004 intuition/intuition.h: *799
LeftEdge          short int in struct NewWindow
         +0x0000  intuition/intuition.h: *976
LeftEdge          short int in struct ExtNewWindow
         +0x0000  intuition/intuition.h: *1046
LeftEdge          short int in struct Screen +0x0008 intuition/screens.h: *103
LeftEdge          short int in struct NewScreen
         +0x0000  intuition/screens.h: *312
LeftEdge          short int in struct ExtNewScreen
         +0x0000  intuition/screens.h: *348
Length            unsigned long int in struct QueryHeader
         +0x000c  graphics/displayinfo.h: *47
LibList           struct List(size 0x000e bytes) in struct ExecBase
         +0x017a  exec/execbase.h: *90
LibNode           struct Library(size 0x0022 bytes) in struct ExecBase
         +0x0000  exec/execbase.h: *37
LibNode           struct Library(size 0x0022 bytes) in struct GfxBase
         +0x0000  graphics/gfxbase.h: *27
LibNode           struct Library(size 0x0022 bytes) in struct IntuitionBase
         +0x0000  intuition/intuitionbase.h: *70
LibNode           struct Library(size 0x0022 bytes) in struct ExpansionBase
         +0x0000  libraries/expansionbase.h: *48
Library           structure tag size 0x0022 exec/libraries.h: *35
                  exec/devices.h: 27
                  exec/execbase.h: 37
                  dos/dosextens.h: 229, 237
                  devices/prtbase.h: 52
                  graphics/gfxbase.h: 27
                  intuition/intuitionbase.h: 70
                  libraries/expansionbase.h: 48
                  libraries/mathlibrary.h: 23
                  resources/disk.h: 51, 55, 56
                  rexx/rxslib.h: 27
LinePtrn          unsigned short int in struct RastPort
         +0x0022  graphics/rastport.h: *73
List              structure tag size 0x000e exec/lists.h: *22
                  exec/tasks.h: 47
                  exec/ports.h: 35
                  exec/interrupts.h: 39
                  exec/execbase.h: 86, 87, 88, 89, 90, 91, 92, 93,
                        114
                  devices/keymap.h: 42
                  graphics/monitor.h: 50
                  graphics/layers.h: 41
                  graphics/gfxbase.h: 37, 50, 52, 86
                  intuition/classusr.h: 119
                  libraries/expansionbase.h: 54, 55
                  resources/disk.h: 58
                  resources/filesysres.h: 30
                  rexx/storage.h: 201, 202, 203, 204, 205
                  rexx/rexxio.h: 67
                  rexx/rxslib.h: 58, 60, 62, 64, 66
                  workbench/workbench.h: 85
LoadSegBlock      structure tag size 0x0200 devices/hardblocks.h: *189
LocalContextItem  structure tag size 0x0014 libraries/iffparse.h: *83
LocalVar          structure tag size 0x0018 dos/var.h: *27
Lock              struct SignalSemaphore(size 0x002e bytes) in struct Layer
         +0x0048  graphics/clip.h: *52
Lock              struct SignalSemaphore(size 0x002e bytes) in struct
                  Layer_Info
         +0x0018  graphics/layers.h: *40
LockLayersCount char in struct Layer_Info +0x005b graphics/layers.h: *45
LongInt           long int in struct StringInfo
         +0x001c  intuition/intuition.h: *553
LongInt           long int in struct SGWork +0x0022 intuition/sghooks.h: *47
LowMemChkSum      short int in struct ExecBase +0x0024 exec/execbase.h: *42
mc_Bytes          unsigned long int in struct MemChunk
```

```
         +0x0004  exec/memory.h: *24
mc_Next           pointer to struct MemChunk in struct MemChunk
         +0x0000  exec/memory.h: *23
mday              unsigned short int in struct ClockData
         +0x0006  utility/date.h: *23
me_Addr           #define me_Un.meu_Addr  exec/memory.h: *52
me_Length         unsigned long int in struct MemEntry
         +0x0004  exec/memory.h: *47
me_Reqs           #define me_Un.meu_Reqs  exec/memory.h: *51
me_Un             union (no tag) (size 0x0004 bytes) in struct MemEntry
         +0x0000  exec/memory.h: *46
me_un             #define me_Un  exec/memory.h: *50
message           struct IOStdReq(size 0x0030 bytes) in struct narrator_rb
         +0x0000  devices/narrator.h: *94
meu_Addr          pointer to void in union (no tag)  +0x0000 exec/memory.h: *45
meu_Reqs          unsigned long int in union (no tag)
         +0x0000  exec/memory.h: *44
mh_Attributes     unsigned short int in struct MemHeader
         +0x000e  exec/memory.h: *32
mh_First          pointer to struct MemChunk in struct MemHeader
         +0x0010  exec/memory.h: *33
mh_Free           unsigned long int in struct MemHeader
         +0x001c  exec/memory.h: *36
mh_Lower          pointer to void in struct MemHeader +0x0014 exec/memory.h: *34
mh_Node           struct Node(size 0x000e bytes) in struct MemHeader
         +0x0000  exec/memory.h: *31
mh_Upper          pointer to void in struct MemHeader +0x0018 exec/memory.h: *35
min               unsigned short int in struct ClockData
         +0x0002  utility/date.h: *21
min_row           unsigned short int in struct MonitorSpec
         +0x002a  graphics/monitor.h: *37
minterms          array [8] of unsigned char in struct RastPort
         +0x0028  graphics/rastport.h: *75
ml_ME             array [1] of struct MemEntry(size 0x0008 bytes) in struct
                  MemList
         +0x0010  exec/memory.h: *61
ml_Node           struct Node(size 0x000e bytes) in struct MemList
         +0x0000  exec/memory.h: *59
ml_NumEntries     unsigned short int in struct MemList
         +0x000e  exec/memory.h: *60
ml_me             #define ml_ME  exec/memory.h: *64
mlh_Head          pointer to struct MinNode in struct MinList
         +0x0000  exec/lists.h: *34
mlh_Tail          pointer to struct MinNode in struct MinList
         +0x0004  exec/lists.h: *35
mlh_TailPred      pointer to struct MinNode in struct MinList
         +0x0008  exec/lists.h: *36
mln_Pred          pointer to struct MinNode in struct MinNode
         +0x0004  exec/nodes.h: *35
mln_Succ          pointer to struct MinNode in struct MinNode
         +0x0000  exec/nodes.h: *34
mn_Length         unsigned short int in struct Message +0x0012 exec/ports.h: *52
mn_Node           struct Node(size 0x000e bytes) in struct Message
         +0x0000  exec/ports.h: *50
mn_ReplyPort      pointer to struct MsgPort in struct Message
         +0x000e  exec/ports.h: *51
mode              unsigned short int in struct narrator_rb
         +0x0034  devices/narrator.h: *97
monitor_id        unsigned short int in struct GfxBase
         +0x00f2  graphics/gfxbase.h: *69
month             unsigned short int in struct ClockData
         +0x0008  utility/date.h: *24
mouth_rb          structure tag size 0x005c devices/narrator.h: *128
mouths            unsigned char in struct narrator_rb
         +0x0042  devices/narrator.h: *103
mp_Flags          unsigned char in struct MsgPort +0x000e exec/ports.h: *32
```

Include File Cross Reference Page 85

```
mp_MsgList           struct List(size 0x000e bytes) in struct MsgPort
             +0x0014 exec/ports.h: *35
mp_Node              struct Node(size 0x000e bytes) in struct MsgPort
             +0x0000 exec/ports.h: *31
mp_SigBit            unsigned char in struct MsgPort +0x000f exec/ports.h: *33
mp_SigTask           pointer to void in struct MsgPort +0x0010 exec/ports.h: *34
mp_SoftInt           #define mp_SigTask  exec/ports.h: *38
ms_Flags             unsigned short int in struct MonitorSpec
             +0x0018 graphics/monitor.h: *30
ms_LegalView         struct Rectangle(size 0x0008 bytes) in struct MonitorSpec
             +0x0042 graphics/monitor.h: *45
ms_Node              struct ExtendedNode(size 0x0018 bytes) in struct MonitorSpec
             +0x0000 graphics/monitor.h: *29
ms_OpenCount         unsigned short int in struct MonitorSpec
             +0x0030 graphics/monitor.h: *39
ms_Special           pointer to struct SpecialMonitor in struct MonitorSpec
             +0x002c graphics/monitor.h: *38
ms_maxoscan          pointer to function returning long int in struct MonitorSpec
             +0x004a graphics/monitor.h: *46
ms_reserved00        unsigned long int in struct MonitorSpec
             +0x0094 graphics/monitor.h: *52
ms_reserved01        unsigned long int in struct MonitorSpec
             +0x0098 graphics/monitor.h: *53
ms_scale             pointer to function returning long int in struct MonitorSpec
             +0x003a graphics/monitor.h: *42
ms_transform         pointer to function returning long int in struct MonitorSpec
             +0x0032 graphics/monitor.h: *40
ms_translate         pointer to function returning long int in struct MonitorSpec
             +0x0036 graphics/monitor.h: *41
ms_videoscan         pointer to function returning long int in struct MonitorSpec
             +0x004e graphics/monitor.h: *47
ms_xoffset           unsigned short int in struct MonitorSpec
             +0x003e graphics/monitor.h: *43
ms_yoffset           unsigned short int in struct MonitorSpec
             +0x0040 graphics/monitor.h: *44
MAKE_ID              macro (4 arguments)   libraries/iffparse.h: *143
MALE                 #define 0 = 0x00000000  devices/narrator.h: *63
MANUALFO             #define 2 = 0x00000002  devices/narrator.h: *67
MATHIEEERESOURCEF_DBLBAS #define (1<<0) = 0x00000001
                     libraries/mathresource.h: *49
MATHIEEERESOURCEF_DBLTRANS #define (1<<1) = 0x00000002
                     libraries/mathresource.h: *50
MATHIEEERESOURCEF_EXTBAS #define (1<<4) = 0x00000010
                     libraries/mathresource.h: *53
MATHIEEERESOURCEF_EXTTRANS #define (1<<5) = 0x00000020
                     libraries/mathresource.h: *54
MATHIEEERESOURCEF_SGLBAS #define (1<<2) = 0x00000004
                     libraries/mathresource.h: *51
MATHIEEERESOURCEF_SGLTRANS #define (1<<3) = 0x00000008
                     libraries/mathresource.h: *52
MAX                  macro (2 arguments)   clib/macros.h: *15
MAXBODY              #define 0xFFFF = 0x0000ffff  intuition/intuition.h: *514
MAXBYTESPERROW       #define 4096 = 0x00001000  hardware/blit.h: *25
MAXCENT              #define 100 = 0x00000064  devices/narrator.h: *88
MAXFONTMATCHWEIGHT #define 32767 = 0x00007fff  graphics/text.h: *87
MAXFONTNAME          #define 32 = 0x00000020  libraries/diskfont.h: *62, 76
MAXFONTPATH          #define 256 = 0x00000100  libraries/diskfont.h: *28, 31, 38
MAXFREQ              #define 28000 = 0x00006d60  devices/narrator.h: *84
MAXINT               #define 0x7FFFFFFF = 0x7fffffff  dos/dos.h: *45
MAXPITCH             #define 320 = 0x00000140  devices/narrator.h: *82
MAXPOT               #define 0xFFFF = 0x0000ffff  intuition/intuition.h: *515
MAXPUBSCREENNAME #define (139) = 0x0000008b  intuition/screens.h: *394
MAXRATE              #define 400 = 0x00000190  devices/narrator.h: *80
MAXRMARG             #define 15 = 0x0000000f  rexx/storage.h: *121
MAXTABS              #define 80 = 0x00000050  devices/conunit.h: *52, 78
MAXVOL               #define 64 = 0x00000040  devices/narrator.h: *86
```

Include File Cross Reference Page 86

```
MAX_MULTIARGS        #define 128 = 0x00000080  dos/rdargs.h: *124
MAX_TEMPLATE_ITEMS #define 100 = 0x00000064  dos/rdargs.h: *117
MCOMPAT_MIXED        #define 0 = 0x00000000  graphics/displayinfo.h: *126
MCOMPAT_NOBODY       #define -1 = 0xffffffff  graphics/displayinfo.h: *128
MCOMPAT_SELF         #define 1 = 0x00000001  graphics/displayinfo.h: *127
MEMCLEAR             #define (1L << 16) = 0x00010000  rexx/storage.h: *221
MEMF_24BITDMA        #define (1L<<9) = 0x00000200  exec/memory.h: *75
MEMF_ANY             #define (0L) = 0x00000000  exec/memory.h: *70
MEMF_CHIP            #define (1L<<1) = 0x00000002  exec/memory.h: *72
MEMF_CLEAR           #define (1L<<16) = 0x00010000  exec/memory.h: *77
MEMF_FAST            #define (1L<<2) = 0x00000004  exec/memory.h: *73
MEMF_LARGEST         #define (1L<<17) = 0x00020000  exec/memory.h: *78
MEMF_LOCAL           #define (1L<<8) = 0x00000100  exec/memory.h: *74
MEMF_PUBLIC          #define (1L<<0) = 0x00000001  exec/memory.h: *71
MEMF_REVERSE         #define (1L<<18) = 0x00040000  exec/memory.h: *79
MEMF_TOTAL           #define (1L<<19) = 0x00080000  exec/memory.h: *80
MEMMASK              #define 0xFFFFFFF0 = 0xfffffff0  rexx/storage.h: *218
MEMQUANT             #define 16L = 0x00000010  rexx/storage.h: *217
MEMQUICK             #define (1L << 0 ) = 0x00000001  rexx/storage.h: *220
MEM_BLOCKMASK        #define (MEM_BLOCKSIZE-1) = 0x00000007  exec/memory.h: *83
MEM_BLOCKSIZE        #define 8L = 0x00000008  exec/memory.h: *82
MENUCANCEL           #define 0x0002 = 0x00000002  intuition/intuition.h: *767
MENUDOWN             #define (IECODE_RBUTTON) = 0x00000069
                     intuition/intuition.h: *1333
MENUENABLED          #define 0x0001 = 0x00000001  intuition/intuition.h: *77
MENUHELP             #define IDCMP_MENUHELP = 0x01000000
                     intuition/iobsolete.h: *138
MENUHOT              #define 0x0001 = 0x00000001  intuition/intuition.h: *766
MENUNULL             #define 0xFFFF = 0x0000ffff  intuition/intuition.h: *1288
MENUNUM              macro (1 argument)   intuition/intuition.h: *1266
MENUPICK             #define IDCMP_MENUPICK = 0x00000100
                     intuition/iobsolete.h: *122
MENUSTATE            #define WFLG_MENUSTATE = 0x00008000
                     intuition/iobsolete.h: *163
MENUTOGGLE           #define 0x0008 = 0x00000008  intuition/intuition.h: *121
MENUTOGGLED          #define 0x4000 = 0x00004000  intuition/intuition.h: *137
MENUUP               #define (IECODE_RBUTTON | IECODE_UP_PREFIX) = 0x000000e9
                     intuition/intuition.h: *1332
MENUVERIFY           #define IDCMP_MENUVERIFY = 0x00002000
                     intuition/iobsolete.h: *127
MENUWAITING          #define 0x0003 = 0x00000003  intuition/intuition.h: *768
MENU_IMAGE           #define 128 = 0x00000080  libraries/gadtools.h: *139
MENU_USERDATA        macro (1 argument)   libraries/gadtools.h: *176
MIDDLEDOWN           #define (IECODE_MBUTTON) = 0x0000006a
                     intuition/intuition.h: *1334
MIDDLEUP             #define (IECODE_MBUTTON | IECODE_UP_PREFIX) = 0x000000ea
                     intuition/intuition.h: *1335
MIDRAWN              #define 0x0100 = 0x00000100  intuition/intuition.h: *80
MIN                  macro (2 arguments)   clib/macros.h: *16
MINBYTESPERROW       #define 128 = 0x00000080  hardware/blit.h: *24
MINCENT              #define 0 = 0x00000000  devices/narrator.h: *87
MINFREQ              #define 5000 = 0x00001388  devices/narrator.h: *83
MININT               #define 0x80000000 = 0x80000000  dos/dos.h: *46
MINPITCH             #define 65 = 0x00000041  devices/narrator.h: *81
MINRATE              #define 40 = 0x00000028  devices/narrator.h: *79
MINVOL               #define 0 = 0x00000000  devices/narrator.h: *85
MIN_NTSC_ROW         #define 21 = 0x00000015  graphics/monitor.h: *80
MIN_PAL_ROW          #define 29 = 0x0000001d  graphics/monitor.h: *81
MIN_VGA70_ROW        #define 35 = 0x00000023  graphics/monitor.h: *111
MIN_VGA_ROW          #define 29 = 0x0000001d  graphics/monitor.h: *96
MISCNAME             #define "misc.resource"  resources/misc.h: *43
MKBADDR              macro (1 argument)   dos/dos.h: *113
MODELCLASS           #define "modelclass"  intuition/classusr.h: *54
MODE_640             #define 0x8000 = 0x00008000  graphics/display.h: *16
MODE_NEWFILE         #define 1006 = 0x000003ee  dos/dos.h: *31
MODE_OLDFILE         #define 1005 = 0x000003ed  dos/dos.h: *29
```

```
MODE_READWRITE    #define 1004 = 0x000003ec  dos/dos.h: *33
MONITOR_ID_MASK   #define 0xFFFF1000 = 0xffff1000  graphics/displayinfo.h: *145
MONITOR_SPEC_TYPE #define 4 = 0x00000004  graphics/gfxnodes.h: *36
MOUSEBUTTONS      #define IDCMP_MOUSEBUTTONS = 0x00000008
                  intuition/iobsolete.h: *117
MOUSEMOVE         #define IDCMP_MOUSEMOVE = 0x00000010
                  intuition/iobsolete.h: *118
MOUSE_ACCEL       #define (1<<15) = 0x00008000  intuition/preferences.h: *138
MR_ALLOCMISCRESOURCE #define (LIB_BASE) = 0xfffffffa  resources/misc.h: *40
MR_FREEMISCRESOURCE #define (LIB_BASE-LIB_VECTSIZE) = 0xfffffff4
                  resources/misc.h: *41
MR_PARALLELBITS   #define 3 = 0x00000003  resources/misc.h: *35
MR_PARALLELPORT   #define 2 = 0x00000002  resources/misc.h: *33
MR_SERIALBITS     #define 1 = 0x00000001  resources/misc.h: *31
MR_SERIALPORT     #define 0 = 0x00000000  resources/misc.h: *30
MTYPE_APPICON     #define 8 = 0x00000008  workbench/workbench.h: *99
MTYPE_APPMENUITEM #define 9 = 0x00000009  workbench/workbench.h: *100
MTYPE_APPWINDOW   #define 7 = 0x00000007  workbench/workbench.h: *98
MTYPE_CLOSEDOWN   #define 5 = 0x00000005  workbench/workbench.h: *96
MTYPE_COPYEXIT    #define 10 = 0x0000000a  workbench/workbench.h: *101
MTYPE_DISKCHANGE  #define 3 = 0x00000003  workbench/workbench.h: *94
MTYPE_ICONPUT     #define 11 = 0x0000000b  workbench/workbench.h: *102
MTYPE_IOPROC      #define 6 = 0x00000006  workbench/workbench.h: *97
MTYPE_PSTD        #define 1 = 0x00000001  workbench/workbench.h: *92
MTYPE_TIMER       #define 4 = 0x00000004  workbench/workbench.h: *95
MTYPE_TOOLEXIT    #define 2 = 0x00000002  workbench/workbench.h: *93
MULTIPLY_DIMENSIONS #define 0x0080 = 0x00000080  intuition/preferences.h: *251
MUSTDRAW          #define 0x0008 = 0x00000008  graphics/gels.h: *25
MXIDCMP           #define (IDCMP_GADGETDOWN) = 0x00000020
                  libraries/gadtools.h: *72
MXIMAGE           #define (0x0FL) = 0x0000000f  intuition/imageclass.h: *111
MX_KIND           #define 5 = 0x00000005  libraries/gadtools.h: *39
M_ASM             #define ">1"  devices/console.h: *99
M_AWM             #define "?7"  devices/console.h: *100
M_LNM             #define 20 = 0x00000014  devices/console.h: *98
                  devices/conunit.h: 98
Mask              unsigned char in struct RastPort
         +0x0018 graphics/rastport.h: *64
Mask              pointer to unsigned short int in struct BoolInfo
         +0x0002 intuition/intuition.h: *431
MathIEEEBase      structure tag size 0x003c libraries/mathlibrary.h: *21
MathIEEEBase_LibNode struct Library(size 0x0022 bytes) in struct MathIEEEBase
         +0x0000 libraries/mathlibrary.h: *23
MathIEEEBase_TaskCloseLib pointer to function returning int in struct
                  MathIEEEBase
         +0x0038 libraries/mathlibrary.h: *26
MathIEEEBase_TaskOpenLib pointer to function returning int in struct
                  MathIEEEBase
         +0x0034 libraries/mathlibrary.h: *25
MathIEEEBase_reserved array [18] of unsigned char in struct MathIEEEBase
         +0x0022 libraries/mathlibrary.h: *24
MathIEEEResource structure tag size 0x002c libraries/mathresource.h: *35
MathIEEEResource_BaseAddr pointer to unsigned short int in struct
                  MathIEEEResource
         +0x0010 libraries/mathresource.h: *39
MathIEEEResource_DblBasInit pointer to function returning void in struct
                  MathIEEEResource
         +0x0014 libraries/mathresource.h: *40
MathIEEEResource_DblTransInit pointer to function returning void in struct
                  MathIEEEResource
         +0x0018 libraries/mathresource.h: *41
MathIEEEResource_ExtBasInit pointer to function returning void in struct
                  MathIEEEResource
         +0x0024 libraries/mathresource.h: *44
MathIEEEResource_ExtTransInit pointer to function returning void in struct
                  MathIEEEResource
```

```
         +0x0028 libraries/mathresource.h: *45
MathIEEEResource_Flags unsigned short int in struct MathIEEEResource
         +0x000e libraries/mathresource.h: *38
MathIEEEResource_Node struct Node(size 0x000e bytes) in struct
                  MathIEEEResource
         +0x0000 libraries/mathresource.h: *37
MathIEEEResource_SglBasInit pointer to function returning void in struct
                  MathIEEEResource
         +0x001c libraries/mathresource.h: *42
MathIEEEResource_SglTransInit pointer to function returning void in struct
                  MathIEEEResource
         +0x0020 libraries/mathresource.h: *43
MaxChars          short int in struct StringInfo
         +0x000a intuition/intuition.h: *530
MaxCount          short int in struct cprlist +0x0008 graphics/copper.h: *60
MaxCount          short int in struct CopList +0x001e graphics/copper.h: *73
MaxCount          short int in struct AreaInfo +0x0012 graphics/rastport.h: *30
MaxDepth          unsigned short int in struct DimensionInfo
         +0x0010 graphics/displayinfo.h: *95
MaxDisplayColumn unsigned short int in struct GfxBase
         +0x00d6 graphics/gfxbase.h: *57
MaxDisplayRow     unsigned short int in struct GfxBase
         +0x00d4 graphics/gfxbase.h: *56
MaxExtMem         pointer to void in struct ExecBase
         +0x004e exec/execbase.h: *53
MaxHeight         unsigned short int in struct Window
         +0x0016 intuition/intuition.h: *805
MaxHeight         unsigned short int in struct NewWindow
         +0x002c intuition/intuition.h: *1026
MaxHeight         unsigned short int in struct ExtNewWindow
         +0x002c intuition/intuition.h: *1061
MaxLocMem         unsigned long int in struct ExecBase
         +0x003e exec/execbase.h: *49
MaxOScan          struct Rectangle(size 0x0008 bytes) in struct DimensionInfo
         +0x0022 graphics/displayinfo.h: *101
MaxRasterHeight unsigned short int in struct DimensionInfo
         +0x0018 graphics/displayinfo.h: *99
MaxRasterWidth    unsigned short int in struct DimensionInfo
         +0x0016 graphics/displayinfo.h: *98
MaxWidth          unsigned short int in struct Window
         +0x0014 intuition/intuition.h: *805
MaxWidth          unsigned short int in struct NewWindow
         +0x002a intuition/intuition.h: *1026
MaxWidth          unsigned short int in struct ExtNewWindow
         +0x002a intuition/intuition.h: *1061
MaxX              short int in struct Rectangle +0x0004 graphics/gfx.h: *32
MaxX              long int in struct Rect32 +0x0008 graphics/gfx.h: *38
MaxY              short int in struct Rectangle +0x0006 graphics/gfx.h: *32
MaxY              long int in struct Rect32 +0x000c graphics/gfx.h: *38
MeMask            short int in struct VSprite +0x0020 graphics/gels.h: *107
MemChunk          structure tag size 0x0008 exec/memory.h: *22, 23, 33
MemEntry          structure tag size 0x0008 exec/memory.h: *42, 61
MemHeader         structure tag size 0x0020 exec/memory.h: *30
MemList           structure tag size 0x0018 exec/memory.h: *58
MemList           struct List(size 0x000e bytes) in struct ExecBase
         +0x0142 exec/execbase.h: *86
Memory            pointer to unsigned char in struct Remember
         +0x0008 intuition/intuition.h: *1235
Menu              structure tag size 0x001e intuition/intuition.h: *62, 64, 809
MenuHBorder       char in struct Screen +0x0022 intuition/screens.h: *121
MenuItem          structure tag
      size 0x0022 intuition/intuition.h: *69, 90, 92, 108
MenuName          pointer to char in struct Menu
         +0x000e intuition/intuition.h: *68
MenuStrip         pointer to struct Menu in struct Window
         +0x001c intuition/intuition.h: *809
```

Include File Cross Reference Page 89

```
MenuVBorder      char in struct Screen +0x0021 intuition/screens.h: *121
Message          structure tag size 0x0014 exec/ports.h: *49
                 exec/io.h: 21, 30
                 devices/audio.h: 55
                 devices/clipboard.h: 44, 60
                 devices/printer.h: 143, 157
                 dos/dosextens.h: 92, 111, 142
                 graphics/text.h: 92
                 intuition/intuition.h: 679
                 dos/notify.h: 42
                 workbench/startup.h: 28
                 resources/disk.h: 44
                 rexx/storage.h: 100
                 workbench/workbench.h: 127
MessageKey       pointer to struct IntuiMessage in struct Window
         +0x005e intuition/intuition.h: *857
MethodID         unsigned long int in struct (no tag)
         +0x0000 intuition/classusr.h: *32
MethodID         unsigned long int in struct opSet
         +0x0000 intuition/classusr.h: *79
MethodID         unsigned long int in struct opUpdate
         +0x0000 intuition/classusr.h: *89
MethodID         unsigned long int in struct opGet
         +0x0000 intuition/classusr.h: *109
MethodID         unsigned long int in struct opAddTail
         +0x0000 intuition/classusr.h: *118
MethodID         unsigned long int in struct opMember
         +0x0000 intuition/classusr.h: *125
MethodID         unsigned long int in struct gpHitTest
         +0x0000 intuition/gadgetclass.h: *177
MethodID         unsigned long int in struct gpRender
         +0x0000 intuition/gadgetclass.h: *189
MethodID         unsigned long int in struct gpInput
         +0x0000 intuition/gadgetclass.h: *202
MethodID         unsigned long int in struct gpGoInactive
         +0x0000 intuition/gadgetclass.h: *232
MethodID         unsigned long int in struct impFrameBox
         +0x0000 intuition/imageclass.h: *140
MethodID         unsigned long int in struct impDraw
         +0x0000 intuition/imageclass.h: *153
MethodID         unsigned long int in struct impErase
         +0x0000 intuition/imageclass.h: *173
MethodID         unsigned long int in struct impHitTest
         +0x0000 intuition/imageclass.h: *189
Micros           unsigned long int in struct IntuiMessage
         +0x0028 intuition/intuition.h: *708
Micros           unsigned long int in struct IntuitionBase
         +0x004c intuition/intuitionbase.h: *87
MicrosPerLine    unsigned short int in struct GfxBase
         +0x00e8 graphics/gfxbase.h: *65
MinDisplayColumn unsigned short int in struct GfxBase
         +0x00ea graphics/gfxbase.h: *66
MinHeight        short int in struct Window +0x0012 intuition/intuition.h: *804
MinHeight        short int in struct NewWindow
         +0x0028 intuition/intuition.h: *1025
MinHeight        short int in struct ExtNewWindow
         +0x0028 intuition/intuition.h: *1060
MinList          structure tag size 0x000c exec/lists.h: *33
                 exec/semaphores.h: 44
                 exec/execbase.h: 142
                 dos/dosextens.h: 65, 251
                 graphics/layers.h: 39
MinNode          structure tag size 0x0008 exec/nodes.h: *33, 34, 35
                 exec/lists.h: 34, 35, 36
                 exec/semaphores.h: 36
                 dos/dosextens.h: 265
```

Include File Cross Reference Page 90

```
                 utility/hooks.h: 25
                 intuition/classes.h: 69
                 libraries/iffparse.h: 70, 84
MinRasterHeight unsigned short int in struct DimensionInfo
         +0x0014 graphics/displayinfo.h: *97
MinRasterWidth   unsigned short int in struct DimensionInfo
         +0x0012 graphics/displayinfo.h: *96
MinRow           unsigned short int in struct MonitorInfo
         +0x0028 graphics/displayinfo.h: *118
MinWidth         short int in struct Window +0x0010 intuition/intuition.h: *804
MinWidth         short int in struct NewWindow
         +0x0026 intuition/intuition.h: *1025
MinWidth         short int in struct ExtNewWindow
         +0x0026 intuition/intuition.h: *1060
MinX             short int in struct Rectangle +0x0000 graphics/gfx.h: *31
MinX             long int in struct Rect32 +0x0000 graphics/gfx.h: *37
MinY             short int in struct Rectangle +0x0002 graphics/gfx.h: *31
MinY             long int in struct Rect32 +0x0004 graphics/gfx.h: *37
Modes            unsigned short int in struct ViewPort
         +0x0020 graphics/view.h: *52
Modes            unsigned short int in struct View +0x0010 graphics/view.h: *65
Modes            unsigned short int in struct GfxBase
         +0x009e graphics/gfxbase.h: *39
Modes            unsigned long int in struct SGWork
         +0x0010 intuition/sghooks.h: *39
Monitor          pointer to struct MonitorSpec in struct ViewExtra
         +0x001c graphics/view.h: *74
MonitorInfo      structure tag size 0x0058 graphics/displayinfo.h: *109
MonitorList      struct List(size 0x000e bytes) in struct GfxBase
         +0x0180 graphics/gfxbase.h: *86
MonitorListSemaphore pointer to struct SignalSemaphore in struct GfxBase
         +0x0192 graphics/gfxbase.h: *88
MonitorSpec      structure tag size 0x009c graphics/monitor.h: *27
                 graphics/view.h: 74
                 graphics/displayinfo.h: 112
                 graphics/gfxbase.h: 85, 87
MoreFlags        unsigned long int in struct Window
         +0x0084 intuition/intuition.h: *901
MountList        struct List(size 0x000e bytes) in struct ExpansionBase
         +0x004a libraries/expansionbase.h: *55
MouseX           short int in struct IntuiMessage
         +0x0020 intuition/intuition.h: *703
MouseX           short int in struct Window +0x000e intuition/intuition.h: *802
MouseX           short int in struct Screen +0x0012 intuition/screens.h: *106
MouseX           short int in struct IntuitionBase
         +0x0046 intuition/intuitionbase.h: *83
MouseY           short int in struct IntuiMessage
         +0x0022 intuition/intuition.h: *703
MouseY           short int in struct Window +0x000c intuition/intuition.h: *802
MouseY           short int in struct Screen +0x0010 intuition/screens.h: *106
MouseY           short int in struct IntuitionBase
         +0x0044 intuition/intuitionbase.h: *83
Msg              pointer to "UBYTE"  intuition/classusr.h: *34
MsgPort          structure tag size 0x0022 exec/ports.h: *30, 51
                 exec/devices.h: 34
                 exec/semaphores.h: 55
                 devices/conunit.h: 56
                 dos/dosextens.h: 41, 93, 94, 112, 253, 267, 334,
                      364, 411, 447
                 graphics/text.h: 126
                 intuition/intuition.h: 856
                 devices/prtbase.h: 67, 92
                 dos/filehandler.h: 104
                 dos/notify.h: 62, 76
                 workbench/startup.h: 29
                 libraries/commodities.h: 57
                 libraries/iffparse.h: 118, 119
                 rexx/storage.h: 108, 189
                 rexx/rexxio.h: 66
                 rexx/rxslib.h: 55
Mspc             pointer to struct MonitorSpec in struct MonitorInfo
         +0x0010 graphics/displayinfo.h: *112
MutualExclude    long int in struct MenuItem +0x000e intuition/intuition.h: *97
MutualExclude    long int in struct Gadget +0x001e intuition/intuition.h: *253
n                struct ExtendedNode(size 0x0018 bytes) in struct ViewExtra
         +0x0000 graphics/view.h: *72
n                struct ExtendedNode(size 0x0018 bytes) in struct
                 ViewPortExtra
         +0x0000 graphics/view.h: *81
n                pointer to struct bltnode in struct bltnode
         +0x0000 hardware/blit.h: *92
narrator_rb      structure tag size 0x0058 devices/narrator.h: *93, 129
nb_Descr         pointer to char in struct NewBroker
         +0x000a libraries/commodities.h: *52
nb_Flags         short int in struct NewBroker
         +0x0010 libraries/commodities.h: *54
nb_Name          pointer to char in struct NewBroker
         +0x0002 libraries/commodities.h: *50
nb_Port          pointer to struct MsgPort in struct NewBroker
         +0x0014 libraries/commodities.h: *57
nb_Pri           char in struct NewBroker +0x0012 libraries/commodities.h: *55
nb_ReservedChannel short int in struct NewBroker
         +0x0018 libraries/commodities.h: *58
nb_Title         pointer to char in struct NewBroker
         +0x0006 libraries/commodities.h: *51
nb_Unique        short int in struct NewBroker
         +0x000e libraries/commodities.h: *53
nb_Version       char in struct NewBroker +0x0000 libraries/commodities.h: *49
nextLine         pointer to short int in struct GelsInfo
         +0x000a graphics/rastport.h: *48
ng_Flags         unsigned long int in struct NewGadget
         +0x0012 libraries/gadtools.h: *101
ng_GadgetID      unsigned short int in struct NewGadget
         +0x0010 libraries/gadtools.h: *100
ng_GadgetText    pointer to unsigned char in struct NewGadget
         +0x0008 libraries/gadtools.h: *98
ng_Height        short int in struct NewGadget
         +0x0006 libraries/gadtools.h: *97
ng_LeftEdge      short int in struct NewGadget
         +0x0000 libraries/gadtools.h: *96
ng_TextAttr      pointer to struct TextAttr in struct NewGadget
         +0x000c libraries/gadtools.h: *99
ng_TopEdge       short int in struct NewGadget
         +0x0002 libraries/gadtools.h: *96
ng_UserData      pointer to void in struct NewGadget
         +0x001a libraries/gadtools.h: *103
ng_VisualInfo    pointer to void in struct NewGadget
         +0x0016 libraries/gadtools.h: *102
ng_Width         short int in struct NewGadget
         +0x0004 libraries/gadtools.h: *97
nm_Class         unsigned long int in struct NotifyMessage
         +0x0014 dos/notify.h: *43
nm_Code          unsigned short int in struct NotifyMessage
         +0x0018 dos/notify.h: *44
nm_CommKey       pointer to unsigned char in struct NewMenu
         +0x0006 libraries/gadtools.h: *127
nm_DoNotTouch    unsigned long int in struct NotifyMessage
         +0x001e dos/notify.h: *46
nm_DoNotTouch2   unsigned long int in struct NotifyMessage
         +0x0022 dos/notify.h: *47
nm_ExecMessage   struct Message(size 0x0014 bytes) in struct NotifyMessage
         +0x0000 dos/notify.h: *42
```

```
nm_Flags         unsigned short int in struct NewMenu
         +0x000a libraries/gadtools.h: *128
nm_Label         pointer to unsigned char in struct NewMenu
         +0x0002 libraries/gadtools.h: *126
nm_MutualExclude long int in struct NewMenu +0x000c libraries/gadtools.h: *129
nm_NReq          pointer to struct NotifyRequest in struct NotifyMessage
         +0x001a dos/notify.h: *45
nm_Type          unsigned char in struct NewMenu
         +0x0000 libraries/gadtools.h: *125
nm_UserData      pointer to void in struct NewMenu
         +0x0010 libraries/gadtools.h: *130
nm_masks         unsigned short int in struct narrator_rb
         +0x003c devices/narrator.h: *100
norm_hblank      array [2] of unsigned short int in struct copinit
         +0x0058 graphics/copper.h: *99
nr_Flags         unsigned long int in struct NotifyRequest
         +0x000c dos/notify.h: *57
nr_FullName      pointer to unsigned char in struct NotifyRequest
         +0x0004 dos/notify.h: *55
nr_Handler       pointer to struct MsgPort in struct NotifyRequest
         +0x002c dos/notify.h: *76
nr_Msg           struct (no tag) (size 0x0004 bytes) in union (no tag)
         +0x0000 dos/notify.h: *63
nr_MsgCount      unsigned long int in struct NotifyRequest
         +0x0028 dos/notify.h: *75
nr_Name          pointer to unsigned char in struct NotifyRequest
         +0x0000 dos/notify.h: *54
nr_Port          pointer to struct MsgPort in struct (no tag)
         +0x0000 dos/notify.h: *62
nr_Reserved      array [4] of unsigned long int in struct NotifyRequest
         +0x0018 dos/notify.h: *72
nr_Signal        struct (no tag) (size 0x0008 bytes) in union (no tag)
         +0x0000 dos/notify.h: *69
nr_SignalNum     unsigned char in struct (no tag)  +0x0004 dos/notify.h: *67
nr_Task          pointer to struct Task in struct (no tag)
         +0x0000 dos/notify.h: *66
nr_UserData      unsigned long int in struct NotifyRequest
         +0x0008 dos/notify.h: *56
nr_pad           array [3] of unsigned char in struct (no tag)
         +0x0005 dos/notify.h: *68
nr_stuff         union (no tag) (size 0x0008 bytes) in struct NotifyRequest
         +0x0010 dos/notify.h: *70
nreserved        array [4] of unsigned long int in struct GfxBase
         +0x0164 graphics/gfxbase.h: *81
ns_Buff          array [8] of char in struct NexxStr
         +0x0008 rexx/storage.h: *47
ns_Flags         unsigned char in struct NexxStr +0x0006 rexx/storage.h: *45
ns_Hash          unsigned char in struct NexxStr +0x0007 rexx/storage.h: *46
ns_Ivalue        long int in struct NexxStr +0x0000 rexx/storage.h: *43
ns_Length        unsigned short int in struct NexxStr
         +0x0004 rexx/storage.h: *44
num              unsigned short int in struct SimpleSprite
         +0x000a graphics/sprite.h: *26
numDrIPens       #define NUMDRIPENS = 0x00000009  intuition/iobsolete.h: *270
numchan          unsigned char in struct narrator_rb
         +0x0044 devices/narrator.h: *105
nxtlist          pointer to struct CopList in union (no tag)
         +0x0000 graphics/copper.h: *31
NABC             #define 0x8 = 0x00000008  hardware/blit.h: *36
NABNC            #define 0x4 = 0x00000004  hardware/blit.h: *37
NANBC            #define 0x2 = 0x00000002  hardware/blit.h: *38
NANBNC           #define 0x1 = 0x00000001  hardware/blit.h: *39
NATURALFO        #define 0 = 0x00000000  devices/narrator.h: *65
NBU_DUPLICATE    #define 0 = 0x00000000  libraries/commodities.h: *62
NBU_NOTIFY       #define 2 = 0x00000002  libraries/commodities.h: *64
NBU_UNIQUE       #define 1 = 0x00000001  libraries/commodities.h: *63
```

```
NB_VERSION         #define 5 = 0x00000005  libraries/commodities.h: *46
NDB_NEWIORB        #define 0 = 0x00000000  devices/narrator.h: *24
NDB_SYLSYNC        #define 2 = 0x00000002  devices/narrator.h: *26
NDB_WORDSYNC       #define 1 = 0x00000001  devices/narrator.h: *25
NDF_NEWIORB        #define (1 << NDB_NEWIORB) = 0x00000001
                   devices/narrator.h: *29
NDF_SYLSYNC        #define (1 << NDB_SYLSYNC) = 0x00000004
                   devices/narrator.h: *31
NDF_WORDSYNC       #define (1 << NDB_WORDSYNC) = 0x00000002
                   devices/narrator.h: *30
ND_CantAlloc       #define -6 = 0xfffffffa  devices/narrator.h: *41
ND_CentPhonErr     #define -28 = 0xffffffe4  devices/narrator.h: *53
ND_DCentErr        #define -27 = 0xffffffe5  devices/narrator.h: *52
ND_Expunged        #define -9 = 0xfffffff7  devices/narrator.h: *44
ND_FreqErr         #define -25 = 0xffffffe7  devices/narrator.h: *50
ND_MakeBad         #define -4 = 0xfffffffc  devices/narrator.h: *39
ND_ModeErr         #define -24 = 0xffffffe8  devices/narrator.h: *49
ND_NoAudLib        #define -3 = 0xfffffffd  devices/narrator.h: *38
ND_NoMem           #define -2 = 0xfffffffe  devices/narrator.h: *37
ND_NoWrite         #define -8 = 0xfffffff8  devices/narrator.h: *43
ND_PhonErr         #define -20 = 0xffffffec  devices/narrator.h: *45
ND_PitchErr        #define -22 = 0xffffffea  devices/narrator.h: *47
ND_RateErr         #define -21 = 0xffffffeb  devices/narrator.h: *46
ND_SexErr          #define -23 = 0xffffffe9  devices/narrator.h: *48
ND_Unimpl          #define -7 = 0xfffffff9  devices/narrator.h: *42
ND_UnitErr         #define -5 = 0xfffffffb  devices/narrator.h: *40
ND_VolErr          #define -26 = 0xffffffe6  devices/narrator.h: *51
NEWLAYERINFO_CALLED #define 1 = 0x00000001  graphics/layers.h: *51
NEWLOCKS           #define  graphics/clip.h: *32
NEWPREFS           #define IDCMP_NEWPREFS = 0x00004000
                   intuition/iobsolete.h: *128
NEWSIZE            #define IDCMP_NEWSIZE = 0x00000002
                   intuition/iobsolete.h: *115
NG_HIGHLABEL       #define 0x0020 = 0x00000020  libraries/gadtools.h: *117
NM_BARLABEL        #define ((STRPTR)-1)  libraries/gadtools.h: *144
NM_END             #define 0 = 0x00000000  libraries/gadtools.h: *137
NM_FLAGMASK        #define (~(COMMSEQ | ITEMTEXT | HIGHFLAGS)) = 0xffffff39
                   libraries/gadtools.h: *157
NM_ITEM            #define 2 = 0x00000002  libraries/gadtools.h: *135
NM_ITEMDISABLED    #define ITEMENABLED = 0x00000010  libraries/gadtools.h: *153
NM_MENUDISABLED    #define MENUENABLED = 0x00000001  libraries/gadtools.h: *152
NM_SUB             #define 3 = 0x00000003  libraries/gadtools.h: *136
NM_TITLE           #define 1 = 0x00000001  libraries/gadtools.h: *134
NOCAREREFRESH      #define WFLG_NOCAREREFRESH = 0x00020000
                   intuition/iobsolete.h: *165
NOCROSSFILL        #define 0x20 = 0x00000020  graphics/rastport.h: *108
NOISYREQ           #define 0x0004 = 0x00000004  intuition/intuition.h: *188
NOITEM             #define 0x003F = 0x0000003f  intuition/intuition.h: *1286
NOMENU             #define 0x001F = 0x0000001f  intuition/intuition.h: *1285
NOREQBACKFILL      #define 0x0040 = 0x00000040  intuition/intuition.h: *198
NOSUB              #define 0x001F = 0x0000001f  intuition/intuition.h: *1287
NOT                #define !  intuition/intuition.h: *1294
NOTIFY_CLASS       #define 0x40000000 = 0x40000000  dos/notify.h: *33
NOTIFY_CODE        #define 0x1234 = 0x00001234  dos/notify.h: *36
NO_ICON_POSITION   #define (0x80000000) = 0x80000000
                   workbench/workbench.h: *116
NP_Arguments       #define (NP_Dummy + 21) = 0x800003fd  dos/dostags.h: *89
NP_Cli             #define (NP_Dummy + 18) = 0x800003fa  dos/dostags.h: *82
NP_CloseError      #define (NP_Dummy + 9) = 0x800003f1  dos/dostags.h: *63
NP_CloseInput      #define (NP_Dummy + 6) = 0x800003ee  dos/dostags.h: *55
NP_CloseOutput     #define (NP_Dummy + 7) = 0x800003ef  dos/dostags.h: *58
NP_CommandName     #define (NP_Dummy + 20) = 0x800003fc  dos/dostags.h: *87
NP_ConsoleTask     #define (NP_Dummy + 14) = 0x800003f6  dos/dostags.h: *74
NP_CopyVars        #define (NP_Dummy + 17) = 0x800003f9  dos/dostags.h: *80
NP_CurrentDir      #define (NP_Dummy + 10) = 0x800003f2  dos/dostags.h: *66
NP_Dummy           #define (TAG_USER + 1000) = 0x800003e8  dos/dostags.h: *42
```

```
NP_Entry           #define (NP_Dummy + 3) = 0x800003eb  dos/dostags.h: *48
NP_Error           #define (NP_Dummy + 8) = 0x800003f0  dos/dostags.h: *61
NP_ExitCode        #define (NP_Dummy + 24) = 0x80000400  dos/dostags.h: *105
NP_ExitData        #define (NP_Dummy + 25) = 0x80000401  dos/dostags.h: *107
NP_FreeSeglist     #define (NP_Dummy + 2) = 0x800003ea  dos/dostags.h: *45
NP_HomeDir         #define (NP_Dummy + 16) = 0x800003f8  dos/dostags.h: *78
NP_Input           #define (NP_Dummy + 4) = 0x800003ec  dos/dostags.h: *51
NP_Name            #define (NP_Dummy + 12) = 0x800003f4  dos/dostags.h: *70
NP_NotifyOnDeath   #define (NP_Dummy + 22) = 0x800003fe  dos/dostags.h: *98
NP_Output          #define (NP_Dummy + 5) = 0x800003ed  dos/dostags.h: *53
NP_Path            #define (NP_Dummy + 19) = 0x800003fb  dos/dostags.h: *84
NP_Priority        #define (NP_Dummy + 13) = 0x800003f5  dos/dostags.h: *72
NP_Seglist         #define (NP_Dummy + 1) = 0x800003e9  dos/dostags.h: *43
NP_StackSize       #define (NP_Dummy + 11) = 0x800003f3  dos/dostags.h: *68
NP_Synchronous     #define (NP_Dummy + 23) = 0x800003ff  dos/dostags.h: *101
NP_WindowPtr       #define (NP_Dummy + 15) = 0x800003f7  dos/dostags.h: *76
NRB_MAGIC          #define 31 = 0x0000001f  dos/notify.h: *94
NRB_NOTIFY_INITIAL #define 4 = 0x00000004  dos/notify.h: *92
NRB_SEND_MESSAGE   #define 0 = 0x00000000  dos/notify.h: *89
NRB_SEND_SIGNAL    #define 1 = 0x00000001  dos/notify.h: *90
NRB_WAIT_REPLY     #define 3 = 0x00000003  dos/notify.h: *91
NRF_MAGIC          #define 0x80000000 = 0x80000000  dos/notify.h: *86
NRF_NOTIFY_INITIAL #define 16 = 0x00000010  dos/notify.h: *83
NRF_SEND_MESSAGE   #define 1 = 0x00000001  dos/notify.h: *80
NRF_SEND_SIGNAL    #define 2 = 0x00000002  dos/notify.h: *81
NRF_WAIT_REPLY     #define 8 = 0x00000008  dos/notify.h: *82
NR_HANDLER_FLAGS   #define 0xffff0000 = 0xffff0000  dos/notify.h: *97
NSB_BINARY         #define 4 = 0x00000004  rexx/storage.h: *58
NSB_EXT            #define 6 = 0x00000006  rexx/storage.h: *60
NSB_FLOAT          #define 5 = 0x00000005  rexx/storage.h: *59
NSB_KEEP           #define 0 = 0x00000000  rexx/storage.h: *54
NSB_NOTNUM         #define 2 = 0x00000002  rexx/storage.h: *56
NSB_NUMBER         #define 3 = 0x00000003  rexx/storage.h: *57
NSB_SOURCE         #define 7 = 0x00000007  rexx/storage.h: *61
NSB_STRING         #define 1 = 0x00000001  rexx/storage.h: *55
NSF_ALPHA          #define (NSF_NOTNUM | NSF_STRING) = 0x00000006
                   rexx/storage.h: *76
NSF_BINARY         #define (1 << NSB_BINARY) = 0x00000010  rexx/storage.h: *68
NSF_DPNUM          #define (NSF_NUMBER | NSF_FLOAT) = 0x00000028
                   rexx/storage.h: *75
NSF_EXT            #define (1 << NSB_EXT ) = 0x00000040  rexx/storage.h: *70
NSF_FLOAT          #define (1 << NSB_FLOAT ) = 0x00000020  rexx/storage.h: *69
NSF_INTNUM         #define (NSF_NUMBER | NSF_BINARY | NSF_STRING) = 0x0000001a
                   rexx/storage.h: *74
NSF_KEEP           #define (1 << NSB_KEEP ) = 0x00000001  rexx/storage.h: *64
NSF_NOTNUM         #define (1 << NSB_NOTNUM) = 0x00000004  rexx/storage.h: *66
NSF_NUMBER         #define (1 << NSB_NUMBER) = 0x00000008  rexx/storage.h: *67
NSF_OWNED          #define (NSF_SOURCE | NSF_EXT | NSF_KEEP) = 0x000000c1
                   rexx/storage.h: *77
NSF_SOURCE         #define (1 << NSB_SOURCE) = 0x00000080  rexx/storage.h: *71
NSF_STRING         #define (1 << NSB_STRING) = 0x00000002  rexx/storage.h: *65
NSTAG_EXT_VPMODE   #define (TAG_USER | 1) = 0x80000001
                   intuition/screens.h: *290
NS_EXTENDED        #define 0x1000 = 0x00001000  intuition/screens.h: *179
NTSC               #define 1 = 0x00000001  graphics/gfxbase.h: *95
NTSC_MONITOR_ID    #define 0x00011000 = 0x00011000  graphics/displayinfo.h: *148
NTSC_MONITOR_NAME  #define "ntsc.monitor"  graphics/monitor.h: *66
NT_BOOTNODE        #define 16 = 0x00000010  exec/nodes.h: *61
NT_DEATHMESSAGE    #define 19 = 0x00000013  exec/nodes.h: *64
NT_DEVICE          #define 3 = 0x00000003  exec/nodes.h: *48
NT_EXTENDED        #define 255 = 0x000000ff  exec/nodes.h: *67
NT_FONT            #define 12 = 0x0000000c  exec/nodes.h: *57
NT_FREEMSG         #define 6 = 0x00000006  exec/nodes.h: *51
NT_GRAPHICS        #define 18 = 0x00000012  exec/nodes.h: *63
NT_INTERRUPT       #define 2 = 0x00000002  exec/nodes.h: *47
NT_KICKMEM         #define 17 = 0x00000011  exec/nodes.h: *62
```

```
NT_LIBRARY          #define 9 = 0x00000009  exec/nodes.h: *54
NT_MEMORY           #define 10 = 0x0000000a  exec/nodes.h: *55
NT_MESSAGE          #define 5 = 0x00000005  exec/nodes.h: *50
NT_MSGPORT          #define 4 = 0x00000004  exec/nodes.h: *49
NT_PROCESS          #define 13 = 0x0000000d  exec/nodes.h: *58
NT_REPLYMSG         #define 7 = 0x00000007  exec/nodes.h: *52
NT_RESOURCE         #define 8 = 0x00000008  exec/nodes.h: *53
NT_SEMAPHORE        #define 14 = 0x0000000e  exec/nodes.h: *59
NT_SIGNALSEM        #define 15 = 0x0000000f  exec/nodes.h: *60
NT_SOFTINT          #define 11 = 0x0000000b  exec/nodes.h: *56
NT_TASK             #define 1 = 0x00000001  exec/nodes.h: *46
NT_UNKNOWN          #define 0 = 0x00000000  exec/nodes.h: *45
NT_USER             #define 254 = 0x000000fe  exec/nodes.h: *66
NULL                #define 0L = 0x00000000  exec/types.h: *76
NULL_IX             macro (1 argument)   libraries/commodities.h: *207
NUMBERIDCMP         #define (NULL) = 0x00000000  libraries/gadtools.h: *73
NUMBER_KIND         #define 6 = 0x00000006  libraries/gadtools.h: *40
NUMDRIPENS          #define (0x0009) = 0x00000009  intuition/screens.h: *92
NUMSECS             #define 11 = 0x0000000b  devices/trackdisk.h: *40
NUMUNITS            #define 4 = 0x00000004  devices/trackdisk.h: *41
NUM_KINDS           #define 14 = 0x0000000e  libraries/gadtools.h: *49
NWAYIDCMP           #define CYCLEIDCMP = 0x00000040  libraries/gadtools.h: *287
NWAY_KIND           #define CYCLE_KIND = 0x00000007  libraries/gadtools.h: *286
NW_EXTENDED         #define WFLG_NW_EXTENDED = 0x00040000
                    intuition/iobsolete.h: *169
NXADDLEN            #define 9 = 0x00000009  rexx/storage.h: *50
NXTLIST             #define u3.nxtlist  graphics/copper.h: *48
N_TRACTOR           #define 0x20 = 0x00000020  intuition/preferences.h: *187
Name                array [32] of unsigned char in struct NameInfo
             +0x0010 graphics/displayinfo.h: *135
NameInfo            structure tag size 0x0038 graphics/displayinfo.h: *132
NewBroker           structure tag size 0x001a libraries/commodities.h: *48
NewGadget           structure tag size 0x001e libraries/gadtools.h: *94
NewMenu             structure tag size 0x0014 libraries/gadtools.h: *123
NewScreen           structure tag size 0x0020 intuition/screens.h: *310
NewWindow           structure tag size 0x0030 intuition/intuition.h: *974
                    workbench/workbench.h: 45, 53
Next                pointer to struct ClipRect in struct ClipRect
             +0x0000 graphics/clip.h: *66
Next                pointer to struct cprlist in struct cprlist
             +0x0000 graphics/copper.h: *58
Next                pointer to struct CopList in struct CopList
             +0x0000 graphics/copper.h: *65
Next                pointer to struct UCopList in struct UCopList
             +0x0000 graphics/copper.h: *85
Next                pointer to struct ViewPort in struct ViewPort
             +0x0000 graphics/view.h: *43
Next                pointer to struct RasInfo in struct RasInfo
             +0x0000 graphics/view.h: *109
Next                pointer to struct RegionRectangle in struct RegionRectangle
             +0x0000 graphics/regions.h: *25
NextBorder          pointer to struct Border in struct Border
             +0x000c intuition/intuition.h: *605
NextComp            pointer to struct AnimComp in struct AnimComp
             +0x0006 graphics/gels.h: *188
NextGadget          pointer to struct Gadget in struct Gadget
             +0x0000 intuition/intuition.h: *218
NextImage           pointer to struct Image in struct Image
             +0x0010 intuition/intuition.h: *666
NextItem            pointer to struct MenuItem in struct MenuItem
             +0x0000 intuition/intuition.h: *92
NextMenu            pointer to struct Menu in struct Menu
             +0x0000 intuition/intuition.h: *64
NextOb              pointer to struct AnimOb in struct AnimOb
             +0x0000 graphics/gels.h: *208
NextRemember        pointer to struct Remember in struct Remember
```

```
             +0x0000 intuition/intuition.h: *1233
NextScreen          pointer to struct Screen in struct Screen
             +0x0000 intuition/screens.h: *100
NextSelect          unsigned short int in struct MenuItem
             +0x0020 intuition/intuition.h: *113
NextSeq             pointer to struct AnimComp in struct AnimComp
             +0x000e graphics/gels.h: *192
NextText            pointer to struct IntuiText in struct IntuiText
             +0x0010 intuition/intuition.h: *578
NextVSprite         pointer to struct VSprite in struct VSprite
             +0x0000 graphics/gels.h: *78
NextWindow          pointer to struct Window in struct Window
             +0x0000 intuition/intuition.h: *797
NexxStr             structure tag size 0x0010 rexx/storage.h: *42
                    rexx/rxslib.h: 37, 38, 39, 40, 41, 42, 43, 44
Node                structure tag size 0x000e exec/nodes.h: *24, 25, 26
                    exec/libraries.h: 36
                    exec/lists.h: 23, 24, 25
                    exec/tasks.h: 27
                    exec/ports.h: 31, 50
                    exec/memory.h: 31, 59
                    exec/interrupts.h: 25, 34
                    exec/semaphores.h: 42
                    devices/clipboard.h: 37
                    devices/keymap.h: 35, 41
                    graphics/gfxnodes.h: 20, 21
                    intuition/screens.h: 383
                    dos/var.h: 28
                    graphics/graphint.h: 22
                    libraries/configvars.h: 35
                    libraries/diskfont.h: 72
                    libraries/expansionbase.h: 37
                    libraries/mathresource.h: 37
                    resources/filesysres.h: 28, 34
                    rexx/storage.h: 163
Nominal             struct Rectangle(size 0x0008 bytes) in struct DimensionInfo
             +0x001a graphics/displayinfo.h: *100
NormalDPMX          unsigned short int in struct GfxBase
             +0x00dc graphics/gfxbase.h: *61
NormalDPMY          unsigned short int in struct GfxBase
             +0x00de graphics/gfxbase.h: *62
NormalDisplayColumns unsigned short int in struct GfxBase
             +0x00da graphics/gfxbase.h: *59
NormalDisplayInfo pointer to void in struct ColorMap
             +0x0018 graphics/view.h: *126
NormalDisplayRows unsigned short int in struct GfxBase
             +0x00d8 graphics/gfxbase.h: *58
NotAvailable        unsigned short int in struct DisplayInfo
             +0x0010 graphics/displayinfo.h: *53
NotifyMessage       structure tag size 0x0026 dos/notify.h: *41
NotifyRequest       structure tag size 0x0030 dos/notify.h: *45, 53
NumChars            short int in struct StringInfo
             +0x0010 intuition/intuition.h: *535
NumChars            short int in struct SGWork +0x001c intuition/sghooks.h: *45
NumStdSprites       unsigned short int in struct DisplayInfo
             +0x001c graphics/displayinfo.h: *57
o_Class             pointer to struct IClass in struct _Object
             +0x0008 intuition/classes.h: *70
o_Node              struct MinNode(size 0x0008 bytes) in struct _Object
             +0x0000 intuition/classes.h: *69
obs                 pointer to struct ClipRect in struct Layer_Info
             +0x0008 graphics/layers.h: *38
opAddMember         #define opMember  intuition/classusr.h: *123
opAddTail           structure tag size 0x0008 intuition/classusr.h: *117
opGet               structure tag size 0x000c intuition/classusr.h: *108
opMember            structure tag size 0x0008 intuition/classusr.h: *124
```

Include File Cross Reference Page 97

```
opSet            structure tag size 0x000c intuition/classusr.h: *78
opUpdate         structure tag size 0x0010 intuition/classusr.h: *88
opam_Object      pointer to unsigned long int in struct opMember
         +0x0004 intuition/classusr.h: *126
opat_List        pointer to struct List in struct opAddTail
         +0x0004 intuition/classusr.h: *119
opg_AttrID       unsigned long int in struct opGet
         +0x0004 intuition/classusr.h: *110
opg_Storage      pointer to unsigned long int in struct opGet
         +0x0008 intuition/classusr.h: *111
ops_AttrList     pointer to struct TagItem in struct opSet
         +0x0004 intuition/classusr.h: *80
ops_GInfo        pointer to struct GadgetInfo in struct opSet
         +0x0008 intuition/classusr.h: *81
opu_AttrList     pointer to struct TagItem in struct opUpdate
         +0x0004 intuition/classusr.h: *90
opu_Flags        unsigned long int in struct opUpdate
         +0x000c intuition/classusr.h: *95
opu_GInfo        pointer to struct GadgetInfo in struct opUpdate
         +0x0008 intuition/classusr.h: *91
OCLASS           macro (1 argument)   intuition/classes.h: *83
OCTANT1          #define 16 = 0x00000010  hardware/blit.h: *87
OCTANT2          #define 0 = 0x00000000  hardware/blit.h: *86
OCTANT3          #define 8 = 0x00000008  hardware/blit.h: *85
OCTANT4          #define 20 = 0x00000014  hardware/blit.h: *84
OCTANT5          #define 28 = 0x0000001c  hardware/blit.h: *83
OCTANT6          #define 12 = 0x0000000c  hardware/blit.h: *82
OCTANT7          #define 4 = 0x00000004  hardware/blit.h: *81
OCTANT8          #define 24 = 0x00000018  hardware/blit.h: *80
OFFSET           macro (2 arguments)   exec/initializers.h: *16
OFFSET_BEGINING  #define OFFSET_BEGINNING = 0xffffffff  dos/dos.h: *40
OFFSET_BEGINNING #define -1 = 0xffffffff  dos/dos.h: *36
OFFSET_CURRENT   #define 0 = 0x00000000  dos/dos.h: *37
OFFSET_END       #define 1 = 0x00000001  dos/dos.h: *38
OFF_DISPLAY      #define custom.dmacon = BITCLR|DMAF_RASTER;
                 graphics/gfxmacros.h: *24
OFF_SPRITE       #define custom.dmacon = BITCLR|DMAF_SPRITE;
                 graphics/gfxmacros.h: *26
OFF_VBLANK       #define custom.intena = BITCLR|INTF_VERTB;
                 graphics/gfxmacros.h: *29
OKABORT          #define 0x0004 = 0x00000004  intuition/intuition.h: *774
OKCANCEL         #define MENUCANCEL = 0x00000002  intuition/intuition.h: *775
OKIMATE_20       #define 0x09 = 0x00000009  intuition/preferences.h: *201
OKOK             #define MENUHOT = 0x00000001  intuition/intuition.h: *773
OLDDRAWERDATAFILESIZE #define (sizeof(struct OldDrawerData)) = 0x00000038
                 workbench/workbench.h: *50
OM_ADDMEMBER     #define (0x109) = 0x00000109  intuition/classusr.h: *72
OM_ADDTAIL       #define (0x105) = 0x00000105  intuition/classusr.h: *68
OM_DISPOSE       #define (0x102) = 0x00000102  intuition/classusr.h: *65
OM_Dummy         #define (0x100) = 0x00000100  intuition/classusr.h: *63
OM_GET           #define (0x104) = 0x00000104  intuition/classusr.h: *67
OM_NEW           #define (0x101) = 0x00000101  intuition/classusr.h: *64
OM_NOTIFY        #define (0x107) = 0x00000107  intuition/classusr.h: *70
OM_REMMEMBER     #define (0x10A) = 0x0000010a  intuition/classusr.h: *73
OM_REMOVE        #define (0x106) = 0x00000106  intuition/classusr.h: *69
OM_SET           #define (0x103) = 0x00000103  intuition/classusr.h: *66
OM_UPDATE        #define (0x108) = 0x00000108  intuition/classusr.h: *71
ONEDOT           #define 0x2 = 0x00000002  hardware/blit.h: *71
ONE_DOT          #define 0x02 = 0x00000002  graphics/rastport.h: *101
ON_DISPLAY       #define custom.dmacon = BITSET|DMAF_RASTER;
                 graphics/gfxmacros.h: *23
ON_SPRITE        #define custom.dmacon = BITSET|DMAF_SPRITE;
                 graphics/gfxmacros.h: *25
ON_VBLANK        #define custom.intena = BITSET|INTF_VERTB;
                 graphics/gfxmacros.h: *28
OPUF_INTERIM     #define (1<<0) = 0x00000001  intuition/classusr.h: *105
```

Include File Cross Reference Page 98

```
ORDERED_DITHERING #define 0x0000 = 0x00000000  intuition/preferences.h: *255
OSCAN_MAX        #define (3) = 0x00000003  intuition/screens.h: *366
OSCAN_STANDARD   #define (2) = 0x00000002  intuition/screens.h: *365
OSCAN_TEXT       #define (1) = 0x00000001  intuition/screens.h: *364
OSCAN_VIDEO      #define (4) = 0x00000004  intuition/screens.h: *367
OSERR_NOCHIPMEM  #define (4) = 0x00000004  intuition/screens.h: *300
OSERR_NOCHIPS    #define (2) = 0x00000002  intuition/screens.h: *298
OSERR_NOMEM      #define (3) = 0x00000003  intuition/screens.h: *299
OSERR_NOMONITOR  #define (1) = 0x00000001  intuition/screens.h: *297
OSERR_PUBNOTUNIQUE #define (5) = 0x00000005  intuition/screens.h: *301
OSERR_UNKNOWNMODE #define (6) = 0x00000006  intuition/screens.h: *302
OTHER_REFRESH    #define WFLG_OTHER_REFRESH = 0x000000c0
                 intuition/iobsolete.h: *155
OUTSTEP          #define 0x2000 = 0x00002000  graphics/gels.h: *43
OVERLAY          #define 0x0004 = 0x00000004  graphics/gels.h: *24
OVFLAG           #define 0x20 = 0x00000020  hardware/blit.h: *72
Object           typedef ULONG  intuition/classusr.h: *21, 126
OldDrawerData    structure tag size 0x0038 workbench/workbench.h: *44
OldX             short int in struct VSprite +0x0012 graphics/gels.h: *91
OldY             short int in struct VSprite +0x0010 graphics/gels.h: *91
OlderRequest     pointer to struct Requester in struct Requester
         +0x0000 intuition/intuition.h: *148
OpCode           short int in struct CopIns +0x0000 graphics/copper.h: *28
pad              unsigned short int in struct BitMap
         +0x0006 graphics/gfx.h: *54
pad              array [4] of unsigned char in struct DisplayInfo
         +0x0024 graphics/displayinfo.h: *60
pad              array [14] of unsigned char in struct DimensionInfo
         +0x0042 graphics/displayinfo.h: *105
pad              array [36] of unsigned char in struct MonitorInfo
         +0x002c graphics/displayinfo.h: *120
pad0             array [255] of unsigned char in struct CIA
         +0x0001 hardware/cia.h: *33
pad1             char in struct narrator_rb +0x0057 devices/narrator.h: *121
pad1             array [255] of unsigned char in struct CIA
         +0x0101 hardware/cia.h: *35
pad10            array [255] of unsigned char in struct CIA
         +0x0a01 hardware/cia.h: *53
pad11            array [255] of unsigned char in struct CIA
         +0x0b01 hardware/cia.h: *55
pad12            array [255] of unsigned char in struct CIA
         +0x0c01 hardware/cia.h: *57
pad13            array [255] of unsigned char in struct CIA
         +0x0d01 hardware/cia.h: *59
pad14            array [255] of unsigned char in struct CIA
         +0x0e01 hardware/cia.h: *61
pad2             array [255] of unsigned char in struct CIA
         +0x0201 hardware/cia.h: *37
pad2d            unsigned char in struct Custom +0x005a hardware/custom.h: *68
pad3             array [255] of unsigned char in struct CIA
         +0x0301 hardware/cia.h: *39
pad34            array [4] of unsigned short int in struct Custom
         +0x0068 hardware/custom.h: *76
pad3b            array [3] of unsigned short int in struct Custom
         +0x0076 hardware/custom.h: *80
pad4             array [255] of unsigned char in struct CIA
         +0x0401 hardware/cia.h: *41
pad5             array [255] of unsigned char in struct CIA
         +0x0501 hardware/cia.h: *43
pad6             array [255] of unsigned char in struct CIA
         +0x0601 hardware/cia.h: *45
pad7             array [255] of unsigned char in struct CIA
         +0x0701 hardware/cia.h: *47
pad8             array [255] of unsigned char in struct CIA
         +0x0801 hardware/cia.h: *49
pad86            array [1] of unsigned short int in struct Custom
```

Include File Cross Reference Page 99

```
         +0x010e hardware/custom.h: *113
pad9             array [255] of unsigned char in struct CIA
         +0x0901 hardware/cia.h: *51
pb_ChkSum        long int in struct PartitionBlock
         +0x0008 devices/hardblocks.h: *135
pb_DevFlags      unsigned long int in struct PartitionBlock
         +0x0020 devices/hardblocks.h: *140
pb_DriveName     array [32] of unsigned char in struct PartitionBlock
         +0x0024 devices/hardblocks.h: *141
pb_EReserved     array [15] of unsigned long int in struct PartitionBlock
         +0x00c4 devices/hardblocks.h: *145
pb_Environment   array [17] of unsigned long int in struct PartitionBlock
         +0x0080 devices/hardblocks.h: *144
pb_Flags         unsigned long int in struct PartitionBlock
         +0x0014 devices/hardblocks.h: *138
pb_HostID        unsigned long int in struct PartitionBlock
         +0x000c devices/hardblocks.h: *136
pb_ID            unsigned long int in struct PartitionBlock
         +0x0000 devices/hardblocks.h: *133
pb_Next          unsigned long int in struct PartitionBlock
         +0x0010 devices/hardblocks.h: *137
pb_Reserved1     array [2] of unsigned long int in struct PartitionBlock
         +0x0018 devices/hardblocks.h: *139
pb_Reserved2     array [15] of unsigned long int in struct PartitionBlock
         +0x0044 devices/hardblocks.h: *143
pb_SummedLongs   unsigned long int in struct PartitionBlock
         +0x0004 devices/hardblocks.h: *134
pd_Device        struct DeviceData(size 0x0034 bytes) in struct PrinterData
         +0x0000 devices/prtbase.h: *66
pd_Flags         unsigned char in struct PrinterData
         +0x09b6 devices/prtbase.h: *95
pd_Flags1        unsigned char in struct PrinterData
         +0x0aa1 devices/prtbase.h: *100
pd_IORPort       struct MsgPort(size 0x0022 bytes) in struct PrinterData
         +0x0138 devices/prtbase.h: *92
pd_OldStk        array [2048] of unsigned char in struct PrinterData
         +0x01b6 devices/prtbase.h: *94
pd_PBothReady    pointer to function returning int in struct PrinterData
         +0x0068 devices/prtbase.h: *74
pd_PIOR0         #define pd_ior0.pd_p0  devices/prtbase.h: *80
pd_PIOR1         #define pd_ior1.pd_p1  devices/prtbase.h: *88
pd_PWaitEnabled unsigned char in struct PrinterData
         +0x0aa0 devices/prtbase.h: *98
pd_PWrite        pointer to function returning int in struct PrinterData
         +0x0064 devices/prtbase.h: *73
pd_Preferences   struct Preferences(size 0x00e8 bytes) in struct PrinterData
         +0x09b8 devices/prtbase.h: *97
pd_PrintBuf      pointer to unsigned char in struct PrinterData
         +0x0060 devices/prtbase.h: *72
pd_PrinterSegment long int in struct PrinterData +0x0056 devices/prtbase.h: *68
pd_PrinterType   unsigned short int in struct PrinterData
         +0x005a devices/prtbase.h: *69
pd_SIOR0         #define pd_ior0.pd_s0  devices/prtbase.h: *81
pd_SIOR1         #define pd_ior1.pd_s1  devices/prtbase.h: *89
pd_SegmentData   pointer to struct PrinterSegment in struct PrinterData
         +0x005c devices/prtbase.h: *71
pd_Stk           array [4096] of unsigned char in struct PrinterData
         +0x0aa2 devices/prtbase.h: *101
pd_TC            struct Task(size 0x005c bytes) in struct PrinterData
         +0x015a devices/prtbase.h: *93
pd_TIOR          struct timerequest(size 0x0028 bytes) in struct PrinterData
         +0x0110 devices/prtbase.h: *91
pd_Unit          struct MsgPort(size 0x0022 bytes) in struct PrinterData
         +0x0034 devices/prtbase.h: *67
pd_ior0          union (no tag) (size 0x0052 bytes) in struct PrinterData
         +0x006c devices/prtbase.h: *78
```

Include File Cross Reference Page 100

```
pd_ior1          union (no tag) (size 0x0052 bytes) in struct PrinterData
         +0x00be devices/prtbase.h: *86
pd_p0            struct IOExtPar(size 0x003e bytes) in union (no tag)
         +0x0000 devices/prtbase.h: *76
pd_p1            struct IOExtPar(size 0x003e bytes) in union (no tag)
         +0x0000 devices/prtbase.h: *84
pd_pad           unsigned char in struct PrinterData
         +0x09b7 devices/prtbase.h: *96
pd_s0            struct IOExtSer(size 0x0052 bytes) in union (no tag)
         +0x0000 devices/prtbase.h: *77
pd_s1            struct IOExtSer(size 0x0052 bytes) in union (no tag)
         +0x0000 devices/prtbase.h: *85
ped_8BitChars    pointer to pointer to char in struct PrinterExtendedData
         +0x0036 devices/prtbase.h: *155
ped_Close        pointer to function returning void in struct
                 PrinterExtendedData
         +0x0010 devices/prtbase.h: *140
ped_ColorClass   unsigned char in struct PrinterExtendedData
         +0x0015 devices/prtbase.h: *142
ped_Commands     pointer to pointer to pointer to char in struct
                 PrinterExtendedData
         +0x0026 devices/prtbase.h: *150
ped_ConvFunc     pointer to function returning int in struct
                 PrinterExtendedData
         +0x003e devices/prtbase.h: *159
ped_DoSpecial    pointer to function returning int in struct
                 PrinterExtendedData
         +0x002a devices/prtbase.h: *151
ped_Expunge      pointer to function returning void in struct
                 PrinterExtendedData
         +0x0008 devices/prtbase.h: *138
ped_Init         pointer to function returning void in struct
                 PrinterExtendedData
         +0x0004 devices/prtbase.h: *137
ped_MaxColumns   unsigned char in struct PrinterExtendedData
         +0x0016 devices/prtbase.h: *143
ped_MaxXDots     unsigned long int in struct PrinterExtendedData
         +0x001a devices/prtbase.h: *146
ped_MaxYDots     unsigned long int in struct PrinterExtendedData
         +0x001e devices/prtbase.h: *147
ped_NumCharSets unsigned char in struct PrinterExtendedData
         +0x0017 devices/prtbase.h: *144
ped_NumRows      unsigned short int in struct PrinterExtendedData
         +0x0018 devices/prtbase.h: *145
ped_Open         pointer to function returning int in struct
                 PrinterExtendedData
         +0x000c devices/prtbase.h: *139
ped_PrintMode    long int in struct PrinterExtendedData
         +0x003a devices/prtbase.h: *156
ped_PrinterClass unsigned char in struct PrinterExtendedData
         +0x0014 devices/prtbase.h: *141
ped_PrinterName pointer to char in struct PrinterExtendedData
         +0x0000 devices/prtbase.h: *136
ped_Render       pointer to function returning int in struct
                 PrinterExtendedData
         +0x002e devices/prtbase.h: *152
ped_TimeoutSecs long int in struct PrinterExtendedData
         +0x0032 devices/prtbase.h: *153
ped_XDotsInch    unsigned short int in struct PrinterExtendedData
         +0x0022 devices/prtbase.h: *148
ped_YDotsInch    unsigned short int in struct PrinterExtendedData
         +0x0024 devices/prtbase.h: *149
pgx_Container    struct IBox(size 0x0008 bytes) in struct PGX
         +0x0000 intuition/cghooks.h: *72
pgx_NewKnob      struct IBox(size 0x0008 bytes) in struct PGX
         +0x0008 intuition/cghooks.h: *73
```

Include File Cross Reference Page 101

```
pi_BotBuf         pointer to unsigned short int in struct PrtInfo
          +0x0038 devices/prtgfx.h: *49
pi_ColorInt       pointer to union colorEntryin struct PrtInfo
          +0x0018 devices/prtgfx.h: *41
pi_ColorIntSize unsigned short int in struct PrtInfo
          +0x0042 devices/prtgfx.h: *54
pi_ColorMap       pointer to union colorEntryin struct PrtInfo
          +0x0014 devices/prtgfx.h: *40
pi_ColorMapSize unsigned short int in struct PrtInfo
          +0x0040 devices/prtgfx.h: *53
pi_Dest1Int       pointer to union colorEntryin struct PrtInfo
          +0x0020 devices/prtgfx.h: *43
pi_Dest1IntSize unsigned short int in struct PrtInfo
          +0x0046 devices/prtgfx.h: *56
pi_Dest2Int       pointer to union colorEntryin struct PrtInfo
          +0x0024 devices/prtgfx.h: *44
pi_Dest2IntSize unsigned short int in struct PrtInfo
          +0x0048 devices/prtgfx.h: *57
pi_HamBuf         pointer to unsigned short int in struct PrtInfo
          +0x0010 devices/prtgfx.h: *39
pi_HamBufSize     unsigned short int in struct PrtInfo
          +0x003e devices/prtgfx.h: *52
pi_HamInt         pointer to union colorEntryin struct PrtInfo
          +0x001c devices/prtgfx.h: *42
pi_HamIntSize     unsigned short int in struct PrtInfo
          +0x0044 devices/prtgfx.h: *55
pi_PrefsFlags     unsigned short int in struct PrtInfo
          +0x004e devices/prtgfx.h: *61
pi_RowBuf         pointer to unsigned short int in struct PrtInfo
          +0x000c devices/prtgfx.h: *38
pi_RowBufSize     unsigned short int in struct PrtInfo
          +0x003c devices/prtgfx.h: *51
pi_ScaleX         pointer to unsigned short int in struct PrtInfo
          +0x0028 devices/prtgfx.h: *45
pi_ScaleXAlt      pointer to unsigned short int in struct PrtInfo
          +0x002c devices/prtgfx.h: *46
pi_ScaleXAltSize unsigned short int in struct PrtInfo
          +0x004c devices/prtgfx.h: *59
pi_ScaleXSize     unsigned short int in struct PrtInfo
          +0x004a devices/prtgfx.h: *58
pi_TopBuf         pointer to unsigned short int in struct PrtInfo
          +0x0034 devices/prtgfx.h: *48
pi_dmatrix        pointer to unsigned char in struct PrtInfo
          +0x0030 devices/prtgfx.h: *47
pi_ety            short int in struct PrtInfo +0x0068 devices/prtgfx.h: *71
pi_flags          unsigned short int in struct PrtInfo
          +0x0070 devices/prtgfx.h: *75
pi_height         unsigned short int in struct PrtInfo
          +0x005a devices/prtgfx.h: *66
pi_pc             unsigned long int in struct PrtInfo
          +0x005c devices/prtgfx.h: *67
pi_pr             unsigned long int in struct PrtInfo
          +0x0060 devices/prtgfx.h: *68
pi_render         pointer to function returning int in struct PrtInfo
          +0x0000 devices/prtgfx.h: *35
pi_rp             pointer to struct RastPort in struct PrtInfo
          +0x0004 devices/prtgfx.h: *36
pi_special        unsigned long int in struct PrtInfo
          +0x0050 devices/prtgfx.h: *62
pi_temprp         pointer to struct RastPort in struct PrtInfo
          +0x0008 devices/prtgfx.h: *37
pi_tempwidth      unsigned short int in struct PrtInfo
          +0x006e devices/prtgfx.h: *74
pi_threshold      unsigned short int in struct PrtInfo
          +0x006c devices/prtgfx.h: *73
pi_width          unsigned short int in struct PrtInfo
```

Include File Cross Reference Page 102

```
          +0x0058 devices/prtgfx.h: *65
pi_xpos           unsigned short int in struct PrtInfo
          +0x006a devices/prtgfx.h: *72
pi_xstart         unsigned short int in struct PrtInfo
          +0x0054 devices/prtgfx.h: *63
pi_ymod           unsigned short int in struct PrtInfo
          +0x0066 devices/prtgfx.h: *70
pi_ymult          unsigned short int in struct PrtInfo
          +0x0064 devices/prtgfx.h: *69
pi_ystart         unsigned short int in struct PrtInfo
          +0x0056 devices/prtgfx.h: *64
pitch             unsigned short int in struct narrator_rb
          +0x0032 devices/narrator.h: *96
pos               unsigned short int in struct SpriteDef
          +0x0000 hardware/custom.h: *117
posctldata        pointer to unsigned short int in struct SimpleSprite
          +0x0000 graphics/sprite.h: *23
pot0dat           unsigned short int in struct Custom
          +0x0012 hardware/custom.h: *37
pot1dat           unsigned short int in struct Custom
          +0x0014 hardware/custom.h: *38
potgo             unsigned short int in struct Custom
          +0x0034 hardware/custom.h: *53
potinp            unsigned short int in struct Custom
          +0x0016 hardware/custom.h: *39
pow               macro (2 arguments)   libraries/mathffp.h: *46
                  libraries/mathieeedp.h: *46
pr_Arguments      pointer to unsigned char in struct Process
          +0x00cc dos/dosextens.h: *64
pr_CES            long int in struct Process +0x00e0 dos/dosextens.h: *67
pr_CIS            long int in struct Process +0x009c dos/dosextens.h: *50
pr_CLI            long int in struct Process +0x00ac dos/dosextens.h: *54
pr_COS            long int in struct Process +0x00a0 dos/dosextens.h: *51
pr_ConsoleTask    pointer to void in struct Process +0x00a4 dos/dosextens.h: *52
pr_CurrentDir     long int in struct Process +0x0098 dos/dosextens.h: *49
pr_ExitCode       pointer to function returning void in struct Process
          +0x00c4 dos/dosextens.h: *62
pr_ExitData       long int in struct Process +0x00c8 dos/dosextens.h: *63
pr_FileSystemTask pointer to void in struct Process
          +0x00a8 dos/dosextens.h: *53
pr_Flags          long int in struct Process +0x00c0 dos/dosextens.h: *61
pr_GlobVec        pointer to void in struct Process +0x0088 dos/dosextens.h: *45
pr_HomeDir        long int in struct Process +0x00bc dos/dosextens.h: *60
pr_LocalVars      struct MinList(size 0x000c bytes) in struct Process
          +0x00d0 dos/dosextens.h: *65
pr_MsgPort        struct MsgPort(size 0x0022 bytes) in struct Process
          +0x005c dos/dosextens.h: *41
pr_Pad            short int in struct Process +0x007e dos/dosextens.h: *42
pr_PktWait        pointer to void in struct Process +0x00b4 dos/dosextens.h: *56
pr_Result2        long int in struct Process +0x0094 dos/dosextens.h: *48
pr_ReturnAddr     pointer to void in struct Process +0x00b0 dos/dosextens.h: *55
pr_SegList        long int in struct Process +0x0080 dos/dosextens.h: *43
pr_ShellPrivate unsigned long int in struct Process
          +0x00dc dos/dosextens.h: *66
pr_StackBase      long int in struct Process +0x0090 dos/dosextens.h: *47
pr_StackSize      long int in struct Process +0x0084 dos/dosextens.h: *44
pr_Task           struct Task(size 0x005c bytes) in struct Process
          +0x0000 dos/dosextens.h: *40
pr_TaskNum        long int in struct Process +0x008c dos/dosextens.h: *46
pr_WindowPtr      pointer to void in struct Process +0x00b8 dos/dosextens.h: *57
prev              pointer to struct ClipRect in struct ClipRect
          +0x0004 graphics/clip.h: *67
priority          char in struct narrator_rb +0x0056 devices/narrator.h: *120
priority          unsigned short int in struct Layer
          +0x001c graphics/clip.h: *41
ps_NextSegment    unsigned long int in struct PrinterSegment
```

```
         +0x0000 devices/prtbase.h: *163
ps_PED           struct PrinterExtendedData(size 0x0042 bytes) in struct
                 PrinterSegment
         +0x000c devices/prtbase.h: *167
ps_Revision      unsigned short int in struct PrinterSegment
         +0x000a devices/prtbase.h: *166
ps_Version       unsigned short int in struct PrinterSegment
         +0x0008 devices/prtbase.h: *165
ps_runAlert      unsigned long int in struct PrinterSegment
         +0x0004 devices/prtbase.h: *164
psn_Flags        unsigned short int in struct PubScreenNode
         +0x0012 intuition/screens.h: *385
psn_Node         struct Node(size 0x000e bytes) in struct PubScreenNode
         +0x0000 intuition/screens.h: *383
psn_Screen       pointer to struct Screen in struct PubScreenNode
         +0x000e intuition/screens.h: *384
psn_SigBit       unsigned char in struct PubScreenNode
         +0x001c intuition/screens.h: *389
psn_SigTask      pointer to struct Task in struct PubScreenNode
         +0x0018 intuition/screens.h: *388
psn_Size         short int in struct PubScreenNode
         +0x0014 intuition/screens.h: *386
psn_VisitorCount short int in struct PubScreenNode
         +0x0016 intuition/screens.h: *387
PAL              #define 4 = 0x00000004  graphics/gfxbase.h: *97
PALETTEIDCMP     #define (IDCMP_GADGETUP) = 0x00000040
                 libraries/gadtools.h: *75
PALETTE_KIND     #define 8 = 0x00000008  libraries/gadtools.h: *42
PAL_MONITOR_ID   #define 0x00021000 = 0x00021000  graphics/displayinfo.h: *149
PAL_MONITOR_NAME #define "pal.monitor"  graphics/monitor.h: *67
PARALLELNAME     #define "parallel.device"  devices/parallel.h: *91
PARALLEL_PRINTER #define 0x00 = 0x00000000  intuition/preferences.h: *141
PARB_ACKMODE     #define 2 = 0x00000002  devices/parallel.h: *67
PARB_EOFMODE     #define 1 = 0x00000001  devices/parallel.h: *70
PARB_FASTMODE    #define 3 = 0x00000003  devices/parallel.h: *62
PARB_RAD_BOOGIE  #define 3 = 0x00000003  devices/parallel.h: *64
PARB_SHARED      #define 5 = 0x00000005  devices/parallel.h: *58
PARB_SLOWMODE    #define 4 = 0x00000004  devices/parallel.h: *60
PARF_ACKMODE     #define (1<<2) = 0x00000004  devices/parallel.h: *68
PARF_EOFMODE     #define (1<<1) = 0x00000002  devices/parallel.h: *71
PARF_FASTMODE    #define (1<<3) = 0x00000008  devices/parallel.h: *63
PARF_RAD_BOOGIE  #define (1<<3) = 0x00000008  devices/parallel.h: *65
PARF_SHARED      #define (1<<5) = 0x00000020  devices/parallel.h: *59
PARF_SLOWMODE    #define (1<<4) = 0x00000010  devices/parallel.h: *61
PA_IGNORE        #define 2 = 0x00000002  exec/ports.h: *44
PA_SIGNAL        #define 0 = 0x00000000  exec/ports.h: *42
PA_SOFTINT       #define 1 = 0x00000001  exec/ports.h: *43
PBFB_BOOTABLE    #define 0 = 0x00000000  devices/hardblocks.h: *150
PBFB_NOMOUNT     #define 1 = 0x00000001  devices/hardblocks.h: *152
PBFF_BOOTABLE    #define 1L = 0x00000001  devices/hardblocks.h: *151
PBFF_NOMOUNT     #define 2L = 0x00000002  devices/hardblocks.h: *153
PCC_4COLOR       #define 0x04 = 0x00000004  devices/prtbase.h: *120
PCC_ADDITIVE     #define 0x08 = 0x00000008  devices/prtbase.h: *121
PCC_BGR          #define 0x0A = 0x0000000a  devices/prtbase.h: *123
PCC_BGRW         #define 0x0C = 0x0000000c  devices/prtbase.h: *125
PCC_BGR_WB       #define 0x0B = 0x0000000b  devices/prtbase.h: *124
PCC_BW           #define 0x01 = 0x00000001  devices/prtbase.h: *116
PCC_MULTI_PASS   #define 0x10 = 0x00000010  devices/prtbase.h: *133
PCC_WB           #define 0x09 = 0x00000009  devices/prtbase.h: *122
PCC_YMC          #define 0x02 = 0x00000002  devices/prtbase.h: *117
PCC_YMCB         #define 0x04 = 0x00000004  devices/prtbase.h: *119
PCC_YMC_BW       #define 0x03 = 0x00000003  devices/prtbase.h: *118
PCMBLACK         #define 3 = 0x00000003  devices/prtgfx.h: *22
PCMBLUE          #define PCMYELLOW = 0x00000000  devices/prtgfx.h: *23
PCMCYAN          #define 2 = 0x00000002  devices/prtgfx.h: *21
PCMGREEN         #define PCMMAGENTA = 0x00000001  devices/prtgfx.h: *24
```

```
PCMMAGENTA        #define 1 = 0x00000001  devices/prtgfx.h: *20
PCMRED            #define PCMCYAN = 0x00000002  devices/prtgfx.h: *25
PCMWHITE          #define PCMBLACK = 0x00000003  devices/prtgfx.h: *26
PCMYELLOW         #define 0 = 0x00000000  devices/prtgfx.h: *19
PDCMD_QUERY       #define (CMD_NONSTD) = 0x00000009  devices/parallel.h: *93
PDCMD_SETPARAMS   #define (CMD_NONSTD+1) = 0x0000000a  devices/parallel.h: *94
PDERR_BADDIMENSION #define 4 = 0x00000004  devices/printer.h: *207
PDERR_BUFFERMEMORY #define 7 = 0x00000007  devices/printer.h: *210
PDERR_CANCEL      #define 1 = 0x00000001  devices/printer.h: *204
PDERR_DIMENSIONOVFLOW #define 5 = 0x00000005  devices/printer.h: *208
PDERR_INTERNALMEMORY #define 6 = 0x00000006  devices/printer.h: *209
PDERR_INVERTHAM   #define 3 = 0x00000003  devices/printer.h: *206
PDERR_NOERR       #define 0 = 0x00000000  devices/printer.h: *203
PDERR_NOTGRAPHICS #define 2 = 0x00000002  devices/printer.h: *205
PDERR_TOOKCONTROL #define 8 = 0x00000008  devices/printer.h: *218
PF2PRI            #define 0x40 = 0x00000040  graphics/display.h: *20
PFA_FINE_SCROLL   #define 0xF = 0x0000000f  graphics/display.h: *27
PFBA              #define 0x0040 = 0x00000040  graphics/view.h: *93
PFB_FINE_SCROLL_SHIFT #define 4 = 0x00000004  graphics/display.h: *28
PFL               pointer to function returning "LONG"   libraries/commodities.h: *8
0
PF_ACTION         #define 3 = 0x00000003  exec/ports.h: *41
PF_FINE_SCROLL_MASK #define 0xF = 0x0000000f  graphics/display.h: *29
PGA_BORDERLESS    #define PGA_Borderless = 0x80031002
                  intuition/iobsolete.h: *216
PGA_Borderless    #define (PGA_Dummy + 0x0002) = 0x80031002
                  intuition/gadgetclass.h: *104
PGA_Dummy         #define (TAG_USER + 0x31000) = 0x80031000
                  intuition/gadgetclass.h: *101
PGA_FREEDOM       #define PGA_Freedom = 0x80031001  intuition/iobsolete.h: *215
PGA_Freedom       #define (PGA_Dummy + 0x0001) = 0x80031001
                  intuition/gadgetclass.h: *102
PGA_HORIZBODY     #define PGA_HorizBody = 0x80031004
                  intuition/iobsolete.h: *218
PGA_HORIZPOT      #define PGA_HorizPot = 0x80031003
                  intuition/iobsolete.h: *217
PGA_HorizBody     #define (PGA_Dummy + 0x0004) = 0x80031004
                  intuition/gadgetclass.h: *106
PGA_HorizPot      #define (PGA_Dummy + 0x0003) = 0x80031003
                  intuition/gadgetclass.h: *105
PGA_NewLook       #define (PGA_Dummy + 0x000A) = 0x8003100a
                  intuition/gadgetclass.h: *113
PGA_TOP           #define PGA_Top = 0x80031009  intuition/iobsolete.h: *223
PGA_TOTAL         #define PGA_Total = 0x80031007  intuition/iobsolete.h: *221
PGA_Top           #define (PGA_Dummy + 0x0009) = 0x80031009
                  intuition/gadgetclass.h: *111
PGA_Total         #define (PGA_Dummy + 0x0007) = 0x80031007
                  intuition/gadgetclass.h: *109
PGA_VERTBODY      #define PGA_VertBody = 0x80031006
                  intuition/iobsolete.h: *220
PGA_VERTPOT       #define PGA_VertPot = 0x80031005  intuition/iobsolete.h: *219
PGA_VISIBLE       #define PGA_Visible = 0x80031008  intuition/iobsolete.h: *222
PGA_VertBody      #define (PGA_Dummy + 0x0006) = 0x80031006
                  intuition/gadgetclass.h: *108
PGA_VertPot       #define (PGA_Dummy + 0x0005) = 0x80031005
                  intuition/gadgetclass.h: *107
PGA_Visible       #define (PGA_Dummy + 0x0008) = 0x80031008
                  intuition/gadgetclass.h: *110
PGX               structure tag size 0x0010 intuition/cghooks.h: *71
PI                #define ((float) 3.141592653589793)  libraries/mathffp.h: *16
                  libraries/mathieeedp.h: 14
PI2               #define (PI/((double)2))  libraries/mathffp.h: *19
                  libraries/mathieeedp.h: *19
PI4               #define (PI/((double)4))  libraries/mathffp.h: *20
                  libraries/mathieeedp.h: *20
PICA              #define 0x000 = 0x00000000  intuition/preferences.h: *159
```

Include File Cross Reference

Page 105

```
PIXEL_DIMENSIONS #define 0x0040 = 0x00000040  intuition/preferences.h: *250
PLACETEXT_ABOVE  #define 0x0004 = 0x00000004  libraries/gadtools.h: *113
PLACETEXT_BELOW  #define 0x0008 = 0x00000008  libraries/gadtools.h: *114
PLACETEXT_IN     #define 0x0010 = 0x00000010  libraries/gadtools.h: *115
PLACETEXT_LEFT   #define 0x0001 = 0x00000001  libraries/gadtools.h: *111
PLACETEXT_RIGHT  #define 0x0002 = 0x00000002  libraries/gadtools.h: *112
PLANEPTR         pointer to "UBYTE"  graphics/gfx.h: *46, 55
PLNCNTMSK        #define 0x7 = 0x00000007  graphics/display.h: *17
PLNCNTSHFT       #define 12 = 0x0000000c  graphics/display.h: *19
PMB_ASM          #define (M_LNM+1) = 0x00000015  devices/conunit.h: *50, 98
PMB_AWM          #define (PMB_ASM+1) = 0x00000016  devices/conunit.h: *51, 98
POINTERSIZE      #define (1 + 16 + 1) * 2 = 0x00000024
                 intuition/preferences.h: *30, 61
POINTREL         #define 0x0001 = 0x00000001  intuition/intuition.h: *181
POPPUBSCREEN     #define 0x0002 = 0x00000002  intuition/screens.h: *398
POTGONAME        #define "potgo.resource"  resources/potgo.h: *14
PPCB_COLOR       #define 1 = 0x00000001  devices/prtbase.h: *107
PPCB_GFX         #define 0 = 0x00000000  devices/prtbase.h: *105
PPCF_COLOR       #define 0x2 = 0x00000002  devices/prtbase.h: *108
PPCF_GFX         #define 0x1 = 0x00000001  devices/prtbase.h: *106
PPC_BWALPHA      #define 0x00 = 0x00000000  devices/prtbase.h: *110
PPC_BWGFX        #define 0x01 = 0x00000001  devices/prtbase.h: *111
PPC_COLORALPHA   #define 0x02 = 0x00000002  devices/prtbase.h: *112
PPC_COLORGFX     #define 0x03 = 0x00000003  devices/prtbase.h: *113
PRB_CLOSEINPUT   #define 3 = 0x00000003  dos/dosextens.h: *79
PRB_CLOSEOUTPUT  #define 4 = 0x00000004  dos/dosextens.h: *81
PRB_FREEARGS     #define 5 = 0x00000005  dos/dosextens.h: *83
PRB_FREECLI      #define 2 = 0x00000002  dos/dosextens.h: *77
PRB_FREECURRDIR  #define 1 = 0x00000001  dos/dosextens.h: *75
PRB_FREESEGLIST  #define 0 = 0x00000000  dos/dosextens.h: *73
PRD_DUMPRPORT    #define (CMD_NONSTD+2) = 0x0000000b  devices/printer.h: *33
PRD_PRTCOMMAND   #define (CMD_NONSTD+1) = 0x0000000a  devices/printer.h: *32
PRD_QUERY        #define (CMD_NONSTD+3) = 0x0000000c  devices/printer.h: *34
PRD_RAWWRITE     #define (CMD_NONSTD+0) = 0x00000009  devices/printer.h: *31
PREDRAWN         #define 0x0002 = 0x00000002  intuition/intuition.h: *186
PRF_CLOSEINPUT   #define 8 = 0x00000008  dos/dosextens.h: *80
PRF_CLOSEOUTPUT  #define 16 = 0x00000010  dos/dosextens.h: *82
PRF_FREEARGS     #define 32 = 0x00000020  dos/dosextens.h: *84
PRF_FREECLI      #define 4 = 0x00000004  dos/dosextens.h: *78
PRF_FREECURRDIR  #define 2 = 0x00000002  dos/dosextens.h: *76
PRF_FREESEGLIST  #define 1 = 0x00000001  dos/dosextens.h: *74
PRIMARY_CLIP     #define 0 = 0x00000000  devices/clipboard.h: *57
PROPBORDERLESS   #define 0x0008 = 0x00000008  intuition/intuition.h: *506
PROPGADGET       #define GTYP_PROPGADGET = 0x00000003
                 intuition/iobsolete.h: *106
PROPGCLASS       #define "propgclass"  intuition/classusr.h: *48
PROPNEWLOOK      #define 0x0010 = 0x00000010  intuition/intuition.h: *510
PROTO_MONITOR_ID #define 0x00051000 = 0x00051000  graphics/displayinfo.h: *214
PSNF_PRIVATE     #define (0x0001) = 0x00000001  intuition/screens.h: *392
PTermArray0      unsigned long int in struct IOPArray
         +0x0000 devices/parallel.h: *20
PTermArray1      unsigned long int in struct IOPArray
         +0x0004 devices/parallel.h: *21
PUBLICSCREEN     #define 0x0002 = 0x00000002  intuition/screens.h: *162
P_ANY            #define 0x80 = 0x00000080  dos/dosasl.h: *124
P_BUFSIZE        #define 256 = 0x00000100  devices/prtbase.h: *62
P_CLASS          #define 0x88 = 0x00000088  dos/dosasl.h: *132
P_NOT            #define 0x85 = 0x00000085  dos/dosasl.h: *129
P_NOTCLASS       #define 0x87 = 0x00000087  dos/dosasl.h: *131
P_NOTEND         #define 0x86 = 0x00000086  dos/dosasl.h: *130
P_OLDSTKSIZE     #define 0x0800 = 0x00000800  devices/prtbase.h: *60, 94
P_OREND          #define 0x84 = 0x00000084  dos/dosasl.h: *128
P_ORNEXT         #define 0x83 = 0x00000083  dos/dosasl.h: *127
P_ORSTART        #define 0x82 = 0x00000082  dos/dosasl.h: *126
P_REPBEG         #define 0x89 = 0x00000089  dos/dosasl.h: *133
P_REPEND         #define 0x8A = 0x0000008a  dos/dosasl.h: *134
```

Include File Cross Reference

Page 106

```
P_SAFESIZE       #define 128 = 0x00000080  devices/prtbase.h: *63
P_SINGLE         #define 0x81 = 0x00000081  dos/dosasl.h: *125
P_STKSIZE        #define 0x1000 = 0x00001000  devices/prtbase.h: *61, 101
P_STOP           #define 0x8B = 0x0000008b  dos/dosasl.h: *135
PaletteRange     unsigned short int in struct DisplayInfo
         +0x001e graphics/displayinfo.h: *58
PaperLength      unsigned short int in struct Preferences
         +0x00b2 intuition/preferences.h: *99
PaperSize        unsigned short int in struct Preferences
         +0x00b0 intuition/preferences.h: *98
PaperType        unsigned short int in struct Preferences
         +0x00b4 intuition/preferences.h: *100
ParErr_BufTooBig #define 2 = 0x00000002  devices/parallel.h: *97
ParErr_DevBusy   #define 1 = 0x00000001  devices/parallel.h: *96
ParErr_InitErr   #define 7 = 0x00000007  devices/parallel.h: *102
ParErr_InvParam  #define 3 = 0x00000003  devices/parallel.h: *98
ParErr_LineErr   #define 4 = 0x00000004  devices/parallel.h: *99
ParErr_NotOpen   #define 5 = 0x00000005  devices/parallel.h: *100
ParErr_PortReset #define 6 = 0x00000006  devices/parallel.h: *101
Parent           pointer to struct Window in struct Window
         +0x0042 intuition/intuition.h: *844
PartitionBlock   structure tag size 0x0100 devices/hardblocks.h: *132
PenHeight        short int in struct RastPort +0x0032 graphics/rastport.h: *77
PenWidth         short int in struct RastPort +0x0030 graphics/rastport.h: *76
Pens             array [2] of unsigned char in struct StringExtend
         +0x0004 intuition/sghooks.h: *22
PixelSpeed       unsigned short int in struct DisplayInfo
         +0x001a graphics/displayinfo.h: *56
PlaneOnOff       unsigned char in struct Image
         +0x000f intuition/intuition.h: *660
PlaneOnOff       char in struct VSprite +0x0039 graphics/gels.h: *136
PlanePick        unsigned char in struct Image
         +0x000e intuition/intuition.h: *660
PlanePick        char in struct VSprite +0x0038 graphics/gels.h: *135
Planes           array [8] of pointer to unsigned char in struct BitMap
         +0x0008 graphics/gfx.h: *55
Point            typedef struct tPoint  graphics/gfx.h: *44
                 graphics/displayinfo.h: 55, 59, 113, 114
Pointer          pointer to unsigned short int in struct Window
         +0x004a intuition/intuition.h: *849
PointerMatrix    array [36] of unsigned short int in struct Preferences
         +0x001c intuition/preferences.h: *61
PointerTicks     unsigned short int in struct Preferences
         +0x006c intuition/preferences.h: *67
PortList         struct List(size 0x000e bytes) in struct ExecBase
         +0x0188 exec/execbase.h: *91
PowerSupplyFrequency unsigned char in struct ExecBase
         +0x0213 exec/execbase.h: *112
Preferences      structure tag size 0x00e8 intuition/preferences.h: *44
                 devices/prtbase.h: 97
Prev             pointer to struct RegionRectangle in struct RegionRectangle
         +0x0004 graphics/regions.h: *25
PrevBuffer       pointer to unsigned char in struct SGWork
         +0x000c intuition/sghooks.h: *38
PrevComp         pointer to struct AnimComp in struct AnimComp
         +0x000a graphics/gels.h: *189
PrevOb           pointer to struct AnimOb in struct AnimOb
         +0x0004 graphics/gels.h: *208
PrevSeq          pointer to struct AnimComp in struct AnimComp
         +0x0012 graphics/gels.h: *193
PrevVSprite      pointer to struct VSprite in struct VSprite
         +0x0004 graphics/gels.h: *79
PrintAspect      unsigned short int in struct Preferences
         +0x00aa intuition/preferences.h: *93
PrintDensity     unsigned char in struct Preferences
         +0x00e0 intuition/preferences.h: *120
```

Include File Cross Reference Page 107

```
PrintFlags        unsigned short int in struct Preferences
          +0x00da intuition/preferences.h: *117
PrintImage        unsigned short int in struct Preferences
          +0x00a8 intuition/preferences.h: *92
PrintLeftMargin unsigned short int in struct Preferences
          +0x00a4 intuition/preferences.h: *90
PrintMaxHeight    unsigned short int in struct Preferences
          +0x00de intuition/preferences.h: *119
PrintMaxWidth     unsigned short int in struct Preferences
          +0x00dc intuition/preferences.h: *118
PrintPitch        unsigned short int in struct Preferences
          +0x009e intuition/preferences.h: *87
PrintQuality      unsigned short int in struct Preferences
          +0x00a0 intuition/preferences.h: *88
PrintRightMargin unsigned short int in struct Preferences
          +0x00a6 intuition/preferences.h: *91
PrintShade        unsigned short int in struct Preferences
          +0x00ac intuition/preferences.h: *94
PrintSpacing      unsigned short int in struct Preferences
          +0x00a2 intuition/preferences.h: *89
PrintThreshold    short int in struct Preferences
          +0x00ae intuition/preferences.h: *95
PrintXOffset      unsigned char in struct Preferences
          +0x00e1 intuition/preferences.h: *121
PrinterData       structure tag (size 0x1aa2 bytes)in struct PrinterData
                  devices/prtbase.h: *65
PrinterExtendedData structure tag size 0x0042 devices/prtbase.h: *135, 167
PrinterFilename array [30] of unsigned char in struct Preferences
          +0x0080 intuition/preferences.h: *84
PrinterPort       unsigned char in struct Preferences
          +0x0001 intuition/preferences.h: *50
PrinterSegment    structure tag size 0x004e devices/prtbase.h: *71, 162
PrinterType       unsigned short int in struct Preferences
          +0x007e intuition/preferences.h: *83
Process           structure tag size 0x00e4 dos/dosextens.h: *39
PropInfo          structure tag size 0x0016 intuition/intuition.h: *452
PropertyFlags     unsigned long int in struct DisplayInfo
          +0x0012 graphics/displayinfo.h: *54
PrtInfo           structure tag size 0x0072 devices/prtgfx.h: *34
PtrHeight         char in struct Window +0x004e intuition/intuition.h: *850
PtrWidth          char in struct Window +0x004f intuition/intuition.h: *851
PubScreenNode     structure tag size 0x001d intuition/screens.h: *382
QUME_LP_20        #define 0x0A = 0x0000000a  intuition/preferences.h: *202
Qualifier         unsigned short int in struct IntuiMessage
          +0x001a intuition/intuition.h: *690
Quantum           unsigned short int in struct ExecBase
          +0x0120 exec/execbase.h: *67
QueryHeader       structure tag
      size 0x0010 graphics/displayinfo.h: *42, 52, 94, 111, 134
ra_Buff           array [8] of char in struct RexxArg
          +0x0008 rexx/storage.h: *92
ra_Flags          unsigned char in struct RexxArg +0x0006 rexx/storage.h: *90
ra_Hash           unsigned char in struct RexxArg +0x0007 rexx/storage.h: *91
ra_Length         unsigned short int in struct RexxArg
          +0x0004 rexx/storage.h: *89
ra_Size           long int in struct RexxArg +0x0000 rexx/storage.h: *88
rate              unsigned short int in struct narrator_rb
          +0x0030 devices/narrator.h: *95
ratioh            long int in struct MonitorSpec +0x001a graphics/monitor.h: *31
ratiov            long int in struct MonitorSpec +0x001e graphics/monitor.h: *32
rdb_AutoParkSeconds unsigned long int in struct RigidDiskBlock
          +0x0094 devices/hardblocks.h: *83
rdb_BadBlockList unsigned long int in struct RigidDiskBlock
          +0x0018 devices/hardblocks.h: *60
rdb_BlockBytes    unsigned long int in struct RigidDiskBlock
          +0x0010 devices/hardblocks.h: *57
```

Include File Cross Reference Page 108

```
rdb_ChkSum        long int in struct RigidDiskBlock
          +0x0008 devices/hardblocks.h: *55
rdb_ControllerProduct array [16] of char in struct RigidDiskBlock
          +0x00c4 devices/hardblocks.h: *90
rdb_ControllerRevision array [4] of char in struct RigidDiskBlock
          +0x00d4 devices/hardblocks.h: *91
rdb_ControllerVendor array [8] of char in struct RigidDiskBlock
          +0x00bc devices/hardblocks.h: *89
rdb_CylBlocks     unsigned long int in struct RigidDiskBlock
          +0x0090 devices/hardblocks.h: *82
rdb_Cylinders     unsigned long int in struct RigidDiskBlock
          +0x0040 devices/hardblocks.h: *67
rdb_DiskProduct array [16] of char in struct RigidDiskBlock
          +0x00a8 devices/hardblocks.h: *87
rdb_DiskRevision array [4] of char in struct RigidDiskBlock
          +0x00b8 devices/hardblocks.h: *88
rdb_DiskVendor    array [8] of char in struct RigidDiskBlock
          +0x00a0 devices/hardblocks.h: *86
rdb_DriveInit     unsigned long int in struct RigidDiskBlock
          +0x0024 devices/hardblocks.h: *63
rdb_FileSysHeaderList unsigned long int in struct RigidDiskBlock
          +0x0020 devices/hardblocks.h: *62
rdb_Flags         unsigned long int in struct RigidDiskBlock
          +0x0014 devices/hardblocks.h: *58
rdb_Heads         unsigned long int in struct RigidDiskBlock
          +0x0048 devices/hardblocks.h: *69
rdb_HiCylinder    unsigned long int in struct RigidDiskBlock
          +0x008c devices/hardblocks.h: *81
rdb_HostID        unsigned long int in struct RigidDiskBlock
          +0x000c devices/hardblocks.h: *56
rdb_ID            unsigned long int in struct RigidDiskBlock
          +0x0000 devices/hardblocks.h: *53
rdb_Interleave    unsigned long int in struct RigidDiskBlock
          +0x004c devices/hardblocks.h: *70
rdb_LoCylinder    unsigned long int in struct RigidDiskBlock
          +0x0088 devices/hardblocks.h: *80
rdb_Park          unsigned long int in struct RigidDiskBlock
          +0x0050 devices/hardblocks.h: *71
rdb_PartitionList unsigned long int in struct RigidDiskBlock
          +0x001c devices/hardblocks.h: *61
rdb_RDBBlocksHi unsigned long int in struct RigidDiskBlock
          +0x0084 devices/hardblocks.h: *79
rdb_RDBBlocksLo unsigned long int in struct RigidDiskBlock
          +0x0080 devices/hardblocks.h: *78
rdb_ReducedWrite unsigned long int in struct RigidDiskBlock
          +0x0064 devices/hardblocks.h: *74
rdb_Reserved1     array [6] of unsigned long int in struct RigidDiskBlock
          +0x0028 devices/hardblocks.h: *65
rdb_Reserved2     array [3] of unsigned long int in struct RigidDiskBlock
          +0x0054 devices/hardblocks.h: *72
rdb_Reserved3     array [5] of unsigned long int in struct RigidDiskBlock
          +0x006c devices/hardblocks.h: *76
rdb_Reserved4     array [2] of unsigned long int in struct RigidDiskBlock
          +0x0098 devices/hardblocks.h: *84
rdb_Reserved5     array [10] of unsigned long int in struct RigidDiskBlock
          +0x00d8 devices/hardblocks.h: *92
rdb_Sectors       unsigned long int in struct RigidDiskBlock
          +0x0044 devices/hardblocks.h: *68
rdb_StepRate      unsigned long int in struct RigidDiskBlock
          +0x0068 devices/hardblocks.h: *75
rdb_SummedLongs unsigned long int in struct RigidDiskBlock
          +0x0004 devices/hardblocks.h: *54
rdb_WritePreComp unsigned long int in struct RigidDiskBlock
          +0x0060 devices/hardblocks.h: *73
rec_FH            long int in struct RecordLock +0x0000 dos/record.h: *30
rec_Length        unsigned long int in struct RecordLock
```

Include File Cross Reference Page 109

```
        +0x0008 dos/record.h: *32
rec_Mode        unsigned long int in struct RecordLock
        +0x000c dos/record.h: *33
rec_Offset      unsigned long int in struct RecordLock
        +0x0004 dos/record.h: *31
refptr          unsigned short int in struct Custom
        +0x0028 hardware/custom.h: *47
reserved        array [4] of unsigned char in struct Layer
        +0x0018 graphics/clip.h: *40
reserved        long int in struct ClipRect +0x0020 graphics/clip.h: *72
reserved        array [8] of unsigned char in struct RastPort
        +0x005c graphics/rastport.h: *89
reserved        array [2] of unsigned long int in struct DisplayInfo
        +0x0028 graphics/displayinfo.h: *61
reserved        array [2] of unsigned long int in struct DimensionInfo
        +0x0050 graphics/displayinfo.h: *106
reserved        array [2] of unsigned long int in struct MonitorInfo
        +0x0050 graphics/displayinfo.h: *121
reserved        array [2] of unsigned long int in struct NameInfo
        +0x0030 graphics/displayinfo.h: *136
reserved1       unsigned long int in struct Layer +0x007a graphics/clip.h: *54
reserved1       pointer to function returning int in struct SpecialMonitor
        +0x001e graphics/monitor.h: *147
reserved1       unsigned char in struct ColorMap +0x0011 graphics/view.h: *123
reserved2       array [18] of unsigned char in struct Layer
        +0x008a graphics/clip.h: *58
reserved2       pointer to function returning int in struct SpecialMonitor
        +0x0022 graphics/monitor.h: *148
reserved2       unsigned short int in struct ColorMap
        +0x0012 graphics/view.h: *124
reserved3       pointer to function returning int in struct SpecialMonitor
        +0x0026 graphics/monitor.h: *149
rf_ArgList      pointer to struct WBArg in struct FileRequester
        +0x0024 libraries/asl.h: *77
rf_Dir          pointer to char in struct FileRequester
        +0x0008 libraries/asl.h: *68
rf_File         pointer to char in struct FileRequester
        +0x0004 libraries/asl.h: *67
rf_Height       short int in struct FileRequester +0x001c libraries/asl.h: *74
rf_LeftEdge     short int in struct FileRequester +0x0016 libraries/asl.h: *73
rf_NumArgs      long int in struct FileRequester +0x0020 libraries/asl.h: *76
rf_Pat          pointer to char in struct FileRequester
        +0x0034 libraries/asl.h: *81
rf_Reserved1    pointer to void in struct FileRequester
        +0x0000 libraries/asl.h: *66
rf_Reserved2    unsigned long int in struct FileRequester
        +0x000c libraries/asl.h: *69
rf_Reserved3    unsigned char in struct FileRequester
        +0x0010 libraries/asl.h: *70
rf_Reserved4    unsigned char in struct FileRequester
        +0x0011 libraries/asl.h: *71
rf_Reserved5    pointer to void in struct FileRequester
        +0x0012 libraries/asl.h: *72
rf_Reserved6    short int in struct FileRequester +0x001e libraries/asl.h: *75
rf_Reserved7    pointer to void in struct FileRequester
        +0x002c libraries/asl.h: *79
rf_Reserved8    pointer to void in struct FileRequester
        +0x0030 libraries/asl.h: *80
rf_TopEdge      short int in struct FileRequester +0x0018 libraries/asl.h: *73
rf_UserData     pointer to void in struct FileRequester
        +0x0028 libraries/asl.h: *78
rf_Width        short int in struct FileRequester +0x001a libraries/asl.h: *74
rightmost       short int in struct GelsInfo +0x0018 graphics/rastport.h: *52
rl_COMMAND      pointer to struct NexxStr in struct RxsLib
        +0x0050 rexx/rxslib.h: *41
rl_CTABLE       pointer to unsigned char in struct RxsLib
```

Include File Cross Reference Page 110

```
        +0x0078 rexx/rxslib.h: *52
rl_Chunk        long int in struct RxsLib +0x0038 rexx/rxslib.h: *35
rl_ClipList     struct List(size 0x000e bytes) in struct RxsLib
        +0x00c8 rexx/rxslib.h: *62
rl_DOSBase      pointer to void in struct RxsLib +0x0028 rexx/rxslib.h: *31
rl_FALSE        pointer to struct NexxStr in struct RxsLib
        +0x0044 rexx/rxslib.h: *38
rl_Flags        unsigned char in struct RxsLib +0x0022 rexx/rxslib.h: *28
rl_IeeeDPBase   pointer to void in struct RxsLib +0x002c rexx/rxslib.h: *32
rl_LibList      struct List(size 0x000e bytes) in struct RxsLib
        +0x00b8 rexx/rxslib.h: *60
rl_MaxNest      long int in struct RxsLib +0x003c rexx/rxslib.h: *36
rl_MsgList      struct List(size 0x000e bytes) in struct RxsLib
        +0x00d8 rexx/rxslib.h: *64
rl_NIL          long int in struct RxsLib +0x0034 rexx/rxslib.h: *34
rl_NULL         pointer to struct NexxStr in struct RxsLib
        +0x0040 rexx/rxslib.h: *37
rl_Node         struct Library(size 0x0022 bytes) in struct RxsLib
        +0x0000 rexx/rxslib.h: *27
rl_Notice       pointer to unsigned char in struct RxsLib
        +0x007c rexx/rxslib.h: *53
rl_NumClip      short int in struct RxsLib +0x00d6 rexx/rxslib.h: *63
rl_NumLib       short int in struct RxsLib +0x00c6 rexx/rxslib.h: *61
rl_NumMsg       short int in struct RxsLib +0x00e6 rexx/rxslib.h: *65
rl_NumPgm       short int in struct RxsLib +0x00f6 rexx/rxslib.h: *67
rl_NumTask      short int in struct RxsLib +0x00b6 rexx/rxslib.h: *59
rl_PgmList      struct List(size 0x000e bytes) in struct RxsLib
        +0x00e8 rexx/rxslib.h: *66
rl_REXX         pointer to struct NexxStr in struct RxsLib
        +0x004c rexx/rxslib.h: *40
rl_ReadLock     unsigned short int in struct RxsLib +0x00a2 rexx/rxslib.h: *56
rl_RexxDir      pointer to unsigned char in struct RxsLib
        +0x0074 rexx/rxslib.h: *51
rl_RexxPort     struct MsgPort(size 0x0022 bytes) in struct RxsLib
        +0x0080 rexx/rxslib.h: *55
rl_STDERR       pointer to struct NexxStr in struct RxsLib
        +0x005c rexx/rxslib.h: *44
rl_STDIN        pointer to struct NexxStr in struct RxsLib
        +0x0054 rexx/rxslib.h: *42
rl_STDOUT       pointer to struct NexxStr in struct RxsLib
        +0x0058 rexx/rxslib.h: *43
rl_SegList      long int in struct RxsLib +0x0030 rexx/rxslib.h: *33
rl_Shadow       unsigned char in struct RxsLib +0x0023 rexx/rxslib.h: *29
rl_StackSize    long int in struct RxsLib +0x0070 rexx/rxslib.h: *50
rl_SysBase      pointer to void in struct RxsLib +0x0024 rexx/rxslib.h: *30
rl_TRUE         pointer to struct NexxStr in struct RxsLib
        +0x0048 rexx/rxslib.h: *39
rl_TaskList     struct List(size 0x000e bytes) in struct RxsLib
        +0x00a8 rexx/rxslib.h: *58
rl_TaskName     pointer to unsigned char in struct RxsLib
        +0x0064 rexx/rxslib.h: *47
rl_TaskPri      long int in struct RxsLib +0x0068 rexx/rxslib.h: *48
rl_TaskSeg      long int in struct RxsLib +0x006c rexx/rxslib.h: *49
rl_TraceCnt     unsigned short int in struct RxsLib +0x00f8 rexx/rxslib.h: *69
rl_TraceFH      long int in struct RxsLib +0x00a4 rexx/rxslib.h: *57
rl_Version      pointer to unsigned char in struct RxsLib
        +0x0060 rexx/rxslib.h: *45
rl_avail        short int in struct RxsLib +0x00fa rexx/rxslib.h: *70
rm_Action       long int in struct RexxMsg +0x001c rexx/storage.h: *103
rm_Args         array [16] of pointer to unsigned char in struct RexxMsg
        +0x0028 rexx/storage.h: *106
rm_CommAddr     pointer to unsigned char in struct RexxMsg
        +0x006c rexx/storage.h: *109
rm_FileExt      pointer to unsigned char in struct RexxMsg
        +0x0070 rexx/storage.h: *110
rm_LibBase      pointer to void in struct RexxMsg +0x0018 rexx/storage.h: *102
```

```
rm_Node          struct Message(size 0x0014 bytes) in struct RexxMsg
         +0x0000 rexx/storage.h: *100
rm_PassPort      pointer to struct MsgPort in struct RexxMsg
         +0x0068 rexx/storage.h: *108
rm_Result1       long int in struct RexxMsg +0x0020 rexx/storage.h: *104
rm_Result2       long int in struct RexxMsg +0x0024 rexx/storage.h: *105
rm_Stdin         long int in struct RexxMsg +0x0074 rexx/storage.h: *111
rm_Stdout        long int in struct RexxMsg +0x0078 rexx/storage.h: *112
rm_TaskBlock     pointer to void in struct RexxMsg +0x0014 rexx/storage.h: *101
rm_avail         long int in struct RexxMsg +0x007c rexx/storage.h: *113
rmp_Node         struct RexxRsrc(size 0x0020 bytes) in struct RexxMsgPort
         +0x0000 rexx/rexxio.h: *65
rmp_Port         struct MsgPort(size 0x0022 bytes) in struct RexxMsgPort
         +0x0020 rexx/rexxio.h: *66
rmp_ReplyList    struct List(size 0x000e bytes) in struct RexxMsgPort
         +0x0042 rexx/rexxio.h: *67
rn_BootProc      pointer to struct MsgPort in struct RootNode
         +0x002c dos/dosextens.h: *253
rn_CliList       struct MinList(size 0x000c bytes) in struct RootNode
         +0x0020 dos/dosextens.h: *251
rn_ConsoleSegment long int in struct RootNode +0x0004 dos/dosextens.h: *246
rn_FileHandlerSegment long int in struct RootNode +0x001c dos/dosextens.h: *250
rn_Flags         long int in struct RootNode +0x0034 dos/dosextens.h: *255
rn_Info          long int in struct RootNode +0x0018 dos/dosextens.h: *249
rn_RestartSeg    long int in struct RootNode +0x0014 dos/dosextens.h: *248
rn_ShellSegment long int in struct RootNode +0x0030 dos/dosextens.h: *254
rn_TaskArray     long int in struct RootNode +0x0000 dos/dosextens.h: *243
rn_Time          struct DateStamp(size 0x000c bytes) in struct RootNode
         +0x0008 dos/dosextens.h: *247
round            macro (1 argument)   libraries/mathffp.h: *32
                 libraries/mathieeedp.h: *32
rp               pointer to struct RastPort in struct Layer
         +0x000c graphics/clip.h: *38
rr_Arg1          long int in struct RexxRsrc +0x0018 rexx/storage.h: *167
rr_Arg2          long int in struct RexxRsrc +0x001c rexx/storage.h: *168
rr_Base          pointer to void in struct RexxRsrc
         +0x0010 rexx/storage.h: *165
rr_Func          short int in struct RexxRsrc +0x000e rexx/storage.h: *164
rr_Node          struct Node(size 0x000e bytes) in struct RexxRsrc
         +0x0000 rexx/storage.h: *163
rr_Size          long int in struct RexxRsrc +0x0014 rexx/storage.h: *166
rt_ClientID      pointer to void in struct RexxTask
         +0x00ec rexx/storage.h: *193
rt_EndSkip       pointer to void in struct Resident
         +0x0006 exec/resident.h: *23
rt_ErrTrap       pointer to void in struct RexxTask
         +0x00fc rexx/storage.h: *198
rt_Flags         unsigned char in struct Resident +0x000a exec/resident.h: *24
rt_Flags         unsigned char in struct RexxTask +0x00ea rexx/storage.h: *190
rt_Global        array [200] of char in struct RexxTask
         +0x0000 rexx/storage.h: *188
rt_Header1       struct List(size 0x000e bytes) in struct RexxTask
         +0x0104 rexx/storage.h: *201
rt_Header2       struct List(size 0x000e bytes) in struct RexxTask
         +0x0112 rexx/storage.h: *202
rt_Header3       struct List(size 0x000e bytes) in struct RexxTask
         +0x0120 rexx/storage.h: *203
rt_Header4       struct List(size 0x000e bytes) in struct RexxTask
         +0x012e rexx/storage.h: *204
rt_Header5       struct List(size 0x000e bytes) in struct RexxTask
         +0x013c rexx/storage.h: *205
rt_IdString      pointer to char in struct Resident
         +0x0012 exec/resident.h: *29
rt_Init          pointer to void in struct Resident
         +0x0016 exec/resident.h: *30
rt_MatchTag      pointer to struct Resident in struct Resident
```

```
         +0x0002 exec/resident.h: *22
rt_MatchWord     unsigned short int in struct Resident
         +0x0000 exec/resident.h: *21
rt_MsgPkt        pointer to void in struct RexxTask
         +0x00f0 rexx/storage.h: *194
rt_MsgPort       struct MsgPort(size 0x0022 bytes) in struct RexxTask
         +0x00c8 rexx/storage.h: *189
rt_Name          pointer to char in struct Resident
         +0x000e exec/resident.h: *28
rt_Pri           char in struct Resident +0x000d exec/resident.h: *27
rt_RexxPort      pointer to void in struct RexxTask
         +0x00f8 rexx/storage.h: *196
rt_SigBit        char in struct RexxTask +0x00eb rexx/storage.h: *191
rt_StackPtr      pointer to void in struct RexxTask
         +0x0100 rexx/storage.h: *199
rt_TaskID        pointer to void in struct RexxTask
         +0x00f4 rexx/storage.h: *195
rt_Type          unsigned char in struct Resident +0x000c exec/resident.h: *26
rt_Version       unsigned char in struct Resident +0x000b exec/resident.h: *25
RASSIZE          macro (2 arguments)   graphics/gfx.h: *58
RATIO_FIXEDPART  #define 4 = 0x00000004  graphics/monitor.h: *133
RATIO_UNITY      #define (1 << RATIO_FIXEDPART) = 0x00000010
                 graphics/monitor.h: *134
RAWKEY           #define IDCMP_RAWKEY = 0x00000400
                 intuition/iobsolete.h: *124
RC_ERROR         #define 10L = 0x0000000a  rexx/errors.h: *74
RC_FATAL         #define 20L = 0x00000014  rexx/errors.h: *75
RC_OK            #define 0L = 0x00000000  rexx/errors.h: *72
RC_WARN          #define 5L = 0x00000005  rexx/errors.h: *73
RDAB_NOALLOC     #define 1 = 0x00000001  dos/rdargs.h: *108
RDAB_NOPROMPT    #define 2 = 0x00000002  dos/rdargs.h: *110
RDAB_STDIN       #define 0 = 0x00000000  dos/rdargs.h: *106
RDAF_NOALLOC     #define 2 = 0x00000002  dos/rdargs.h: *109
RDAF_NOPROMPT    #define 4 = 0x00000004  dos/rdargs.h: *111
RDAF_STDIN       #define 1 = 0x00000001  dos/rdargs.h: *107
RDA_BufSiz       long int in struct RDArgs +0x0014 dos/rdargs.h: *101
RDA_Buffer       pointer to unsigned char in struct RDArgs
         +0x0010 dos/rdargs.h: *100
RDA_DAList       long int in struct RDArgs +0x000c dos/rdargs.h: *99
RDA_ExtHelp      pointer to unsigned char in struct RDArgs
         +0x0018 dos/rdargs.h: *102
RDA_Flags        long int in struct RDArgs +0x001c dos/rdargs.h: *103
RDA_Source       struct CSource(size 0x000c bytes) in struct RDArgs
         +0x0000 dos/rdargs.h: *98
RDArgs           structure tag size 0x0020 dos/rdargs.h: *97
RDBFB_CTRLRID    #define 5 = 0x00000005  devices/hardblocks.h: *109
RDBFB_DISKID     #define 4 = 0x00000004  devices/hardblocks.h: *107
RDBFB_LAST       #define 0 = 0x00000000  devices/hardblocks.h: *99
RDBFB_LASTLUN    #define 1 = 0x00000001  devices/hardblocks.h: *101
RDBFB_LASTTID    #define 2 = 0x00000002  devices/hardblocks.h: *103
RDBFB_NORESELECT #define 3 = 0x00000003  devices/hardblocks.h: *105
RDBFF_CTRLRID    #define 0x20L = 0x00000020  devices/hardblocks.h: *110
RDBFF_DISKID     #define 0x10L = 0x00000010  devices/hardblocks.h: *108
RDBFF_LAST       #define 0x01L = 0x00000001  devices/hardblocks.h: *100
RDBFF_LASTLUN    #define 0x02L = 0x00000002  devices/hardblocks.h: *102
RDBFF_LASTTID    #define 0x04L = 0x00000004  devices/hardblocks.h: *104
RDBFF_NORESELECT #define 0x08L = 0x00000008  devices/hardblocks.h: *106
RDB_LOCATION_LIMIT #define 16 = 0x00000010  devices/hardblocks.h: *97
RECOVERY_ALERT   #define 0x00000000 = 0x00000000  intuition/intuition.h: *1311
REC_EXCLUSIVE    #define 0 = 0x00000000  dos/record.h: *22
REC_EXCLUSIVE_IMMED #define 1 = 0x00000001  dos/record.h: *23
REC_SHARED       #define 2 = 0x00000002  dos/record.h: *24
REC_SHARED_IMMED #define 3 = 0x00000003  dos/record.h: *25
REFRESHBITS      #define WFLG_REFRESHBITS = 0x000000c0
                 intuition/iobsolete.h: *151
REFRESHWINDOW    #define IDCMP_REFRESHWINDOW = 0x00000004
```

Include File Cross Reference Page 113

```
                intuition/iobsolete.h: *116
REGISTER        #define register  exec/types.h: *23
RELVERIFY       #define GACT_RELVERIFY = 0x00000001
                intuition/iobsolete.h: *69
REPORTMOUSE     #define WFLG_REPORTMOUSE = 0x00000200
                intuition/iobsolete.h: *157
REPORT_INSERT   #define 4 = 0x00000004  dos/dosextens.h: *456
REPORT_LOCK     #define 2 = 0x00000002  dos/dosextens.h: *454
REPORT_STREAM   #define 0 = 0x00000000  dos/dosextens.h: *452
REPORT_TASK     #define 1 = 0x00000001  dos/dosextens.h: *453
REPORT_VOLUME   #define 3 = 0x00000003  dos/dosextens.h: *455
REQACTIVE       #define 0x2000 = 0x00002000  intuition/intuition.h: *204
REQCLEAR        #define IDCMP_REQCLEAR = 0x00001000
                intuition/iobsolete.h: *126
REQGADGET       #define GTYP_REQGADGET = 0x00001000
                intuition/iobsolete.h: *95
REQOFFWINDOW    #define 0x1000 = 0x00001000  intuition/intuition.h: *203
REQSET          #define IDCMP_REQSET = 0x00000080
                intuition/iobsolete.h: *121
REQUEST_A2024   #define 8 = 0x00000008  graphics/monitor.h: *63
REQUEST_NTSC    #define 1 = 0x00000001  graphics/monitor.h: *60
REQUEST_PAL     #define 2 = 0x00000002  graphics/monitor.h: *61
REQUEST_SPECIAL #define 4 = 0x00000004  graphics/monitor.h: *62
REQVERIFY       #define IDCMP_REQVERIFY = 0x00000800
                intuition/iobsolete.h: *125
RESCOUNT        #define 2 = 0x00000002  intuition/intuitionbase.h: *42
RESOURCES_BATTCLOCK_H #define 1 = 0x00000001  resources/battclock.h: *2
RESOURCES_BATTMEMBITSAMIGA_H #define 1 = 0x00000001
                resources/battmembitsamiga.h: *2
RESOURCES_BATTMEMBITSAMIX_H #define 1 = 0x00000001
                resources/battmembitsamix.h: *2
RESOURCES_BATTMEMBITSSHARED_H #define 1 = 0x00000001
                resources/battmembitsshared.h: *2
RESOURCES_BATTMEM_H #define 1 = 0x00000001  resources/battmem.h: *2
RESOURCES_CIA_H  #define  resources/ciabase.h: *2
RESOURCES_DISK_H #define  resources/disk.h: *2
RESOURCES_FILESYSRES_H #define  resources/filesysres.h: *2
RESOURCES_MATHRESOURCE_H #define  libraries/mathresource.h: *2
                resources/mathresource.h: 1
RESOURCES_MISC_H #define  resources/misc.h: *2
RESOURCES_POTGO_H #define  resources/potgo.h: *2
RETURN_ERROR    #define 10 = 0x0000000a  dos/dos.h: *200
RETURN_FAIL     #define 20 = 0x00000014  dos/dos.h: *201
RETURN_OK       #define 0 = 0x00000000  dos/dos.h: *198
RETURN_WARN     #define 5 = 0x00000005  dos/dos.h: *199
REXX_ERRORS_H   #define  rexx/errors.h: *2
REXX_REXXIO_H   #define  rexx/rexxio.h: *2
REXX_RXSLIB_H   #define  rexx/rxslib.h: *2
REXX_STORAGE_H  #define  rexx/storage.h: *2, 1
                rexx/rxslib.h: 15
RIGHTBORDER     #define GACT_RIGHTBORDER = 0x00000010
                intuition/iobsolete.h: *73
RIGHTHIT        #define 8 = 0x00000008  graphics/collide.h: *35
RIGHTIMAGE      #define (0x0CL) = 0x0000000c  intuition/imageclass.h: *108
RINGTRIGGER     #define 0x0001 = 0x00000001  graphics/gels.h: *48
RI_VERSION      #define (1) = 0x00000001  intuition/screens.h: *58
RLFB_CLOSE      #define 7 = 0x00000007  rexx/rxslib.h: *78
RLFB_HALT       #define RTFB_HALT = 0x00000001  rexx/rxslib.h: *75
RLFB_STOP       #define 6 = 0x00000006  rexx/rxslib.h: *77
RLFB_SUSP       #define RTFB_SUSP = 0x00000002  rexx/rxslib.h: *76
RLFB_TRACE      #define RTFB_TRACE = 0x00000000  rexx/rxslib.h: *74
RLFMASK         #define (1<<RLFB_TRACE) | (1<<RLFB_HALT) | (1<<RLFB_SUSP) =
                0x00000007
                rexx/rxslib.h: *80
RMBTRAP         #define WFLG_RMBTRAP = 0x00010000
                intuition/iobsolete.h: *164
```

Include File Cross Reference Page 114

```
RNB_PRIVATE1    #define 1 = 0x00000001  dos/dosextens.h: *260
RNB_WILDSTAR    #define 24 = 0x00000018  dos/dosextens.h: *258
RNF_PRIVATE1    #define 2 = 0x00000002  dos/dosextens.h: *261
RNF_WILDSTAR    #define (1L<<24) = 0x01000000  dos/dosextens.h: *259
ROBOTICFO       #define 1 = 0x00000001  devices/narrator.h: *66
ROOTCLASS       #define "rootclass"  intuition/classusr.h: *42
RPTR            short int  exec/types.h: *51
RP_User         pointer to pointer to void in struct RastPort
         +0x0042 graphics/rastport.h: *85
RPort           pointer to struct RastPort in struct Window
         +0x0032 intuition/intuition.h: *820
RRT_ANY         #define 0 = 0x00000000  rexx/storage.h: *172
RRT_CLIP        #define 5 = 0x00000005  rexx/storage.h: *177
RRT_FILE        #define 3 = 0x00000003  rexx/storage.h: *175
RRT_HOST        #define 4 = 0x00000004  rexx/storage.h: *176
RRT_LIB         #define 1 = 0x00000001  rexx/storage.h: *173
RRT_PORT        #define 2 = 0x00000002  rexx/storage.h: *174
RTC_MATCHWORD   #define 0x4AFC = 0x00004afc  exec/resident.h: *33
RTFB_CLOSE      #define 7 = 0x00000007  rexx/storage.h: *214
RTFB_HALT       #define 1 = 0x00000001  rexx/storage.h: *210
RTFB_SUSP       #define 2 = 0x00000002  rexx/storage.h: *211
RTFB_TCUSE      #define 3 = 0x00000003  rexx/storage.h: *212
RTFB_TRACE      #define 0 = 0x00000000  rexx/storage.h: *209
RTFB_WAIT       #define 6 = 0x00000006  rexx/storage.h: *213
RTF_AFTERDOS    #define (1<<2) = 0x00000004  exec/resident.h: *36
RTF_AUTOINIT    #define (1<<7) = 0x00000080  exec/resident.h: *35
RTF_COLDSTART   #define (1<<0) = 0x00000001  exec/resident.h: *38
RTF_SINGLETASK  #define (1<<1) = 0x00000002  exec/resident.h: *37
RTW_COLDSTART   #define 1 = 0x00000001  exec/resident.h: *43
RTW_NEVER       #define 0 = 0x00000000  exec/resident.h: *42
RUN_EXECUTE     #define -1 = 0xffffffff  dos/dosextens.h: *464
RUN_SYSTEM      #define -2 = 0xfffffffe  dos/dosextens.h: *465
RUN_SYSTEM_ASYNCH #define -3 = 0xfffffffd  dos/dosextens.h: *466
RWindow         pointer to struct Window in struct Requester
         +0x0048 intuition/intuition.h: *172
RXADDCON        #define 0x0A000000 = 0x0a000000  rexx/storage.h: *131
RXADDFH         #define 0x07000000 = 0x07000000  rexx/storage.h: *128
RXADDLIB        #define 0x08000000 = 0x08000000  rexx/storage.h: *129
RXARGMASK       #define 0x0000000F = 0x0000000f  rexx/storage.h: *151
RXBUFFSZ        #define 204 = 0x000000cc  rexx/rexxio.h: *19, 32
RXCLOSE         #define 0x03000000 = 0x03000000  rexx/storage.h: *126
RXCODEMASK      #define 0xFF000000 = 0xff000000  rexx/storage.h: *150
RXCOMM          #define 0x01000000 = 0x01000000  rexx/storage.h: *124
RXFB_NOIO       #define 16 = 0x00000010  rexx/storage.h: *137
RXFB_NONRET     #define 20 = 0x00000014  rexx/storage.h: *141
RXFB_RESULT     #define 17 = 0x00000011  rexx/storage.h: *138
RXFB_STRING     #define 18 = 0x00000012  rexx/storage.h: *139
RXFB_TOKEN      #define 19 = 0x00000013  rexx/storage.h: *140
RXFF_NOIO       #define (1L << RXFB_NOIO ) = 0x00010000  rexx/storage.h: *144
RXFF_NONRET     #define (1L << RXFB_NONRET) = 0x00100000
                rexx/storage.h: *148
RXFF_RESULT     #define (1L << RXFB_RESULT) = 0x00020000
                rexx/storage.h: *145
RXFF_STRING     #define (1L << RXFB_STRING) = 0x00040000
                rexx/storage.h: *146
RXFF_TOKEN      #define (1L << RXFB_TOKEN ) = 0x00080000
                rexx/storage.h: *147
RXFUNC          #define 0x02000000 = 0x02000000  rexx/storage.h: *125
RXIO_APPEND     #define 3 = 0x00000003  rexx/rexxio.h: *40
RXIO_BEGIN      #define -1L = 0xffffffff  rexx/rexxio.h: *45
RXIO_CURR       #define 0L = 0x00000000  rexx/rexxio.h: *46
RXIO_END        #define 1L = 0x00000001  rexx/rexxio.h: *47
RXIO_EXIST      #define -1 = 0xffffffff  rexx/rexxio.h: *36
RXIO_READ       #define 1 = 0x00000001  rexx/rexxio.h: *38
RXIO_STRF       #define 0 = 0x00000000  rexx/rexxio.h: *37
RXIO_WRITE      #define 2 = 0x00000002  rexx/rexxio.h: *39
```

```
RXQUERY          #define 0x04000000 = 0x04000000  rexx/storage.h: *127
RXREMCON         #define 0x0B000000 = 0x0b000000  rexx/storage.h: *132
RXREMLIB         #define 0x09000000 = 0x09000000  rexx/storage.h: *130
RXSCHUNK         #define 1024 = 0x00000400  rexx/rxslib.h: *83
RXSDIR           #define "REXX"  rexx/rxslib.h: *20
RXSNAME          #define "rexxsyslib.library"  rexx/rxslib.h: *19
RXSNEST          #define 32 = 0x00000020  rexx/rxslib.h: *84
RXSSTACK         #define 4096 = 0x00001000  rexx/rxslib.h: *86
RXSTNAME         #define "ARexx"  rexx/rxslib.h: *21
RXSTPRI          #define 0 = 0x00000000  rexx/rxslib.h: *85
RXTCCLS          #define 0x0D000000 = 0x0d000000  rexx/storage.h: *134
RXTCOPN          #define 0x0C000000 = 0x0c000000  rexx/storage.h: *133
RasInfo          structure tag size 0x000c graphics/view.h: *55, 107, 109
RasInfo          pointer to struct RasInfo in struct ViewPort
         +0x0024 graphics/view.h: *55
RasPtr           pointer to char in struct TmpRas
         +0x0000 graphics/rastport.h: *36
RastPort         structure tag (size 0x0064 bytes)in struct impErase
                 devices/prtgfx.h: *36, 37
                 devices/printer.h: 163
                 graphics/clip.h: 38
                 graphics/rastport.h: 56
                 intuition/intuition.h: 820, 833
                 intuition/screens.h: 128
                 intuition/cghooks.h: 37
                 intuition/gadgetclass.h: 191
                 intuition/imageclass.h: 154, 174
RastPort         struct RastPort(size 0x0064 bytes) in struct Screen
         +0x0054 intuition/screens.h: *128
ReadChar         #define FGetC(Input())  dos/stdio.h: *17
ReadChars        macro (2 arguments)   dos/stdio.h: *21
ReadLn           macro (2 arguments)   dos/stdio.h: *22
RecordLock       structure tag size 0x0010 dos/record.h: *29
Rect32           structure tag size 0x0010 graphics/gfx.h: *35
Rectangle        structure tag size 0x0008 graphics/gfx.h: *29
                 graphics/clip.h: 39, 70
                 graphics/monitor.h: 45
                 graphics/view.h: 83
                 graphics/text.h: 169
                 graphics/displayinfo.h: 100, 101, 102, 103, 104,
                           115
                 graphics/regions.h: 26, 31
Red              unsigned short int in struct ColorSpec
         +0x0002 intuition/intuition.h: *1243
Region           structure tag size 0x000c graphics/clip.h: *55, 56, 60
                 graphics/regions.h: 29
RegionRectangle structure tag size 0x0010 graphics/regions.h: *23, 25, 32
RegionRectangle pointer to struct RegionRectangle in struct Region
         +0x0008 graphics/regions.h: *32
RelLeft          short int in struct Requester
         +0x000c intuition/intuition.h: *151
RelTop           short int in struct Requester
         +0x000e intuition/intuition.h: *151
RemBob           macro (1 argument)   graphics/gels.h: *247
Remember         structure tag size 0x000c intuition/intuition.h: *1231, 1233
RememberSize     unsigned long int in struct Remember
         +0x0004 intuition/intuition.h: *1234
ReqBorder        pointer to struct Border in struct Requester
         +0x0014 intuition/intuition.h: *154
ReqCount         short int in struct Window +0x002c intuition/intuition.h: *817
ReqGadget        pointer to struct Gadget in struct Requester
         +0x0010 intuition/intuition.h: *153
ReqImage         pointer to struct Image in struct Requester
         +0x004c intuition/intuition.h: *174
ReqLayer         pointer to struct Layer in struct Requester
         +0x0020 intuition/intuition.h: *161
```

```
ReqPad1          array [32] of unsigned char in struct Requester
         +0x0024 intuition/intuition.h: *163
ReqPad2          array [32] of unsigned char in struct Requester
         +0x0050 intuition/intuition.h: *176
ReqText          pointer to struct IntuiText in struct Requester
         +0x0018 intuition/intuition.h: *155
Requester        structure tag
     size 0x0070 intuition/intuition.h: *146, 148, 813, 815
                 intuition/cghooks.h: 31
ResModules       pointer to void in struct ExecBase
         +0x012c exec/execbase.h: *76
Reserved         unsigned long int in struct BoolInfo
         +0x0006 intuition/intuition.h: *437
Reserved         array [4] of unsigned long int in struct StringExtend
         +0x0014 intuition/sghooks.h: *30
Resident         structure tag size 0x001a exec/resident.h: *20, 22
Resolution       struct tPoint(size 0x0004 bytes) in struct DisplayInfo
         +0x0016 graphics/displayinfo.h: *55
ResourceList     struct List(size 0x000e bytes) in struct ExecBase
         +0x0150 exec/execbase.h: *87
RexxArg          structure tag size 0x0010 rexx/storage.h: *87
RexxMsg          structure tag size 0x0080 rexx/storage.h: *99
RexxMsgPort      structure tag size 0x0050 rexx/rexxio.h: *64
RexxRsrc         structure tag size 0x0020 rexx/storage.h: *162
                 rexx/rexxio.h: 26, 65
RexxTask         structure tag size 0x014a rexx/storage.h: *187
RigidDiskBlock   structure tag size 0x0100 devices/hardblocks.h: *49
RingXTrans       short int in struct AnimOb +0x001e graphics/gels.h: *217
RingYTrans       short int in struct AnimOb +0x001c graphics/gels.h: *217
RootNode         structure tag size 0x0038 dos/dosextens.h: *230, 242
RowSizeChange    char in struct Preferences
         +0x00d8 intuition/preferences.h: *114
Rows             unsigned short int in struct BitMap
         +0x0002 graphics/gfx.h: *51
RxOffset         short int in struct RasInfo +0x0008 graphics/view.h: *111
RxsLib           structure tag size 0x00fc rexx/rxslib.h: *26
RyOffset         short int in struct RasInfo +0x000a graphics/view.h: *111
sampfreq         unsigned short int in struct narrator_rb
         +0x0040 devices/narrator.h: *102
saveClipRects    pointer to struct Region in struct Layer
         +0x0082 graphics/clip.h: *56
sc_Buf           pointer to void in struct IFFStreamCmd
         +0x0004 libraries/iffparse.h: *58
sc_Command       long int in struct IFFStreamCmd
         +0x0000 libraries/iffparse.h: *57
sc_NBytes        long int in struct IFFStreamCmd
         +0x0008 libraries/iffparse.h: *59
scsi_Actual      unsigned long int in struct SCSICmd
         +0x0008 devices/scsidisk.h: *80
scsi_CmdActual   unsigned short int in struct SCSICmd
         +0x0012 devices/scsidisk.h: *83
scsi_CmdLength   unsigned short int in struct SCSICmd
         +0x0010 devices/scsidisk.h: *82
scsi_Command     pointer to unsigned char in struct SCSICmd
         +0x000c devices/scsidisk.h: *81
scsi_Data        pointer to unsigned short int in struct SCSICmd
         +0x0000 devices/scsidisk.h: *74
scsi_Flags       unsigned char in struct SCSICmd
         +0x0014 devices/scsidisk.h: *84
scsi_Length      unsigned long int in struct SCSICmd
         +0x0004 devices/scsidisk.h: *77
scsi_SenseActual unsigned short int in struct SCSICmd
         +0x001c devices/scsidisk.h: *91
scsi_SenseData   pointer to unsigned char in struct SCSICmd
         +0x0016 devices/scsidisk.h: *86
scsi_SenseLength unsigned short int in struct SCSICmd
```

Include File Cross Reference Page 117

```
         +0x001a devices/scsidisk.h: *89
scsi_Status      unsigned char in struct SCSICmd
         +0x0015 devices/scsidisk.h: *85
sec              unsigned short int in struct ClockData
         +0x0000 utility/date.h: *20
seg_Name         array [4] of unsigned char in struct Segment
         +0x000c dos/dosextens.h: *292
seg_Next         long int in struct Segment +0x0000 dos/dosextens.h: *289
seg_Seg          long int in struct Segment +0x0008 dos/dosextens.h: *291
seg_UC           long int in struct Segment +0x0004 dos/dosextens.h: *290
serdat           unsigned short int in struct Custom
         +0x0030 hardware/custom.h: *51
serdatr          unsigned short int in struct Custom
         +0x0018 hardware/custom.h: *40
serper           unsigned short int in struct Custom
         +0x0032 hardware/custom.h: *52
sex              unsigned short int in struct narrator_rb
         +0x0036 devices/narrator.h: *98
sh_List          struct List(size 0x000e bytes) in struct SoftIntList
         +0x0000 exec/interrupts.h: *39
sh_Pad           unsigned short int in struct SoftIntList
         +0x000e exec/interrupts.h: *40
shadowPen        #define SHADOWPEN = 0x00000004  intuition/iobsolete.h: *265
shape            unsigned char in struct mouth_rb
         +0x005a devices/narrator.h: *132
shinePen         #define SHINEPEN = 0x00000003  intuition/iobsolete.h: *264
sin              #define IEEEDPSin  libraries/mathffp.h: *43
                 libraries/mathieeedp.h: *43
sinh             #define IEEEDPSinh  libraries/mathffp.h: *51
                 libraries/mathieeedp.h: *51
sm_ArgList       pointer to struct WBArg in struct WBStartup
         +0x0024 workbench/startup.h: *33
sm_Bids          short int in struct Semaphore +0x0022 exec/semaphores.h: *56
sm_ClipID        long int in struct SatisfyMsg +0x0016 devices/clipboard.h: *62
sm_LockMsg       #define mp_SigTask  exec/semaphores.h: *59
sm_Message       struct Message(size 0x0014 bytes) in struct WBStartup
         +0x0000 workbench/startup.h: *28
sm_Msg           struct Message(size 0x0014 bytes) in struct SatisfyMsg
         +0x0000 devices/clipboard.h: *60
sm_MsgPort       struct MsgPort(size 0x0022 bytes) in struct Semaphore
         +0x0000 exec/semaphores.h: *55
sm_NumArgs       long int in struct WBStartup +0x001c workbench/startup.h: *31
sm_Process       pointer to struct MsgPort in struct WBStartup
         +0x0014 workbench/startup.h: *29
sm_Segment       long int in struct WBStartup +0x0018 workbench/startup.h: *30
sm_ToolWindow    pointer to char in struct WBStartup
         +0x0020 workbench/startup.h: *32
sm_Unit          unsigned short int in struct SatisfyMsg
         +0x0014 devices/clipboard.h: *61
sn_Pred          pointer to struct SrcNode in struct SrcNode
         +0x0004 rexx/storage.h: *229
sn_Ptr           pointer to void in struct SrcNode +0x0008 rexx/storage.h: *230
sn_Size          long int in struct SrcNode +0x000c rexx/storage.h: *231
sn_Succ          pointer to struct SrcNode in struct SrcNode
         +0x0000 rexx/storage.h: *228
sp_Data          pointer to unsigned char in struct StoredProperty
         +0x0004 libraries/iffparse.h: *97
sp_Msg           struct Message(size 0x0014 bytes) in struct StandardPacket
         +0x0000 dos/dosextens.h: *142
sp_Pkt           struct DosPacket(size 0x0030 bytes) in struct StandardPacket
         +0x0014 dos/dosextens.h: *143
sp_Size          long int in struct StoredProperty
         +0x0000 libraries/iffparse.h: *96
spm_Flags        unsigned short int in struct SpecialMonitor
         +0x0018 graphics/monitor.h: *145
spm_Node         struct ExtendedNode(size 0x0018 bytes) in struct
```

Include File Cross Reference Page 118

```
                 SpecialMonitor
         +0x0000 graphics/monitor.h: *144
spr              array [8] of struct SpriteDef(size 0x0008 bytes) in struct
                 Custom
         +0x0140 hardware/custom.h: *121
sprRsrvd         char in struct GelsInfo +0x0000 graphics/rastport.h: *43
sprhstop         unsigned short int in struct Custom
         +0x01d2 hardware/custom.h: *132
sprhstrt         unsigned short int in struct Custom
         +0x01d0 hardware/custom.h: *131
sprpt            array [8] of pointer to void in struct Custom
         +0x0120 hardware/custom.h: *115
sprstop          array [4] of unsigned short int in struct copinit
         +0x0070 graphics/copper.h: *103
sprstrtup        array [32] of unsigned short int in struct copinit
         +0x0014 graphics/copper.h: *97
sqrt             #define IEEEDPSqrt  libraries/mathffp.h: *49
                 libraries/mathieeedp.h: *49
sr_Link          struct MinNode(size 0x0008 bytes) in struct SemaphoreRequest
         +0x0000 exec/semaphores.h: *36
sr_Waiter        pointer to struct Task in struct SemaphoreRequest
         +0x0008 exec/semaphores.h: *37
ss_Link          struct Node(size 0x000e bytes) in struct SignalSemaphore
         +0x0000 exec/semaphores.h: *42
ss_MultipleLink struct SemaphoreRequest(size 0x000c bytes) in struct
                 SignalSemaphore
         +0x001c exec/semaphores.h: *45
ss_NestCount     short int in struct SignalSemaphore
         +0x000e exec/semaphores.h: *43
ss_Owner         pointer to struct Task in struct SignalSemaphore
         +0x0028 exec/semaphores.h: *46
ss_QueueCount    short int in struct SignalSemaphore
         +0x002c exec/semaphores.h: *47
ss_WaitQueue     struct MinList(size 0x000c bytes) in struct SignalSemaphore
         +0x0010 exec/semaphores.h: *44
start            pointer to unsigned short int in struct cprlist
         +0x0004 graphics/copper.h: *59
stat             char in struct bltnode +0x0008 hardware/blit.h: *94
strequ           unsigned short int in struct Custom
         +0x0038 hardware/custom.h: *55
strhor           unsigned short int in struct Custom
         +0x003c hardware/custom.h: *57
strlong          unsigned short int in struct Custom
         +0x003e hardware/custom.h: *58
strvbl           unsigned short int in struct Custom
         +0x003a hardware/custom.h: *56
sync             unsigned char in struct mouth_rb
         +0x005b devices/narrator.h: *133
system_bplcon0   short int in struct GfxBase +0x00a4 graphics/gfxbase.h: *43
SAVEBACK         #define 0x0002 = 0x00000002  graphics/gels.h: *23
SAVEBOB          #define 0x0001 = 0x00000001  graphics/gels.h: *35
SAVEPRESERVE     #define 0x1000 = 0x00001000  graphics/gels.h: *42
SA_AutoScroll    #define (SA_Dummy + 0x0019) = 0x80000039
                 intuition/screens.h: *273
SA_Behind        #define (SA_Dummy + 0x0017) = 0x80000037
                 intuition/screens.h: *269
SA_BitMap        #define (SA_Dummy + 0x000E) = 0x8000002e
                 intuition/screens.h: *229
SA_BlockPen      #define (SA_Dummy + 0x0007) = 0x80000027
                 intuition/screens.h: *210
SA_Colors        #define (SA_Dummy + 0x0009) = 0x80000029
                 intuition/screens.h: *213
SA_DClip         #define (SA_Dummy + 0x0013) = 0x80000033
                 intuition/screens.h: *247
SA_Depth         #define (SA_Dummy + 0x0005) = 0x80000025
                 intuition/screens.h: *206
```

Include File Cross Reference Page 119

```
SA_DetailPen       #define (SA_Dummy + 0x0006) = 0x80000026
                   intuition/screens.h: *208
SA_DisplayID       #define (SA_Dummy + 0x0012) = 0x80000032
                   intuition/screens.h: *243
SA_Dummy           #define (TAG_USER + 32) = 0x80000020
                   intuition/screens.h: *197
SA_ErrorCode       #define (SA_Dummy + 0x000A) = 0x8000002a
                   intuition/screens.h: *218
SA_Font            #define (SA_Dummy + 0x000B) = 0x8000002b
                   intuition/screens.h: *220
SA_FullPalette     #define (SA_Dummy + 0x001B) = 0x8000003b
                   intuition/screens.h: *279
SA_Height          #define (SA_Dummy + 0x0004) = 0x80000024
                   intuition/screens.h: *204
SA_Left            #define (SA_Dummy + 0x0001) = 0x80000021
                   intuition/screens.h: *201
SA_Obsolete1       #define (SA_Dummy + 0x0015) = 0x80000035
                   intuition/screens.h: *263
SA_Overscan        #define (SA_Dummy + 0x0014) = 0x80000034
                   intuition/screens.h: *251
SA_Pens            #define (SA_Dummy + 0x001A) = 0x8000003a
                   intuition/screens.h: *275
SA_PubName         #define (SA_Dummy + 0x000F) = 0x8000002f
                   intuition/screens.h: *233
SA_PubSig          #define (SA_Dummy + 0x0010) = 0x80000030
                   intuition/screens.h: *238
SA_PubTask         #define (SA_Dummy + 0x0011) = 0x80000031
                   intuition/screens.h: *239
SA_Quiet           #define (SA_Dummy + 0x0018) = 0x80000038
                   intuition/screens.h: *271
SA_ShowTitle       #define (SA_Dummy + 0x0016) = 0x80000036
                   intuition/screens.h: *267
SA_SysFont         #define (SA_Dummy + 0x000C) = 0x8000002c
                   intuition/screens.h: *222
SA_Title           #define (SA_Dummy + 0x0008) = 0x80000028
                   intuition/screens.h: *211
SA_Top             #define (SA_Dummy + 0x0002) = 0x80000022
                   intuition/screens.h: *202
SA_Type            #define (SA_Dummy + 0x000D) = 0x8000002d
                   intuition/screens.h: *227
SA_Width           #define (SA_Dummy + 0x0003) = 0x80000023
                   intuition/screens.h: *203
SBUFSIZE_BITS      #define 0x0F = 0x0000000f  intuition/preferences.h: *220
SBUF_1024          #define 0x01 = 0x00000001  intuition/preferences.h: *209
SBUF_16000         #define 0x05 = 0x00000005  intuition/preferences.h: *213
SBUF_2048          #define 0x02 = 0x00000002  intuition/preferences.h: *210
SBUF_4096          #define 0x03 = 0x00000003  intuition/preferences.h: *211
SBUF_512           #define 0x00 = 0x00000000  intuition/preferences.h: *208
SBUF_8000          #define 0x04 = 0x00000004  intuition/preferences.h: *212
SCREENBEHIND       #define 0x0080 = 0x00000080  intuition/screens.h: *173
SCREENHIRES        #define 0x0200 = 0x00000200  intuition/screens.h: *177
SCREENQUIET        #define 0x0100 = 0x00000100  intuition/screens.h: *176
SCREENTYPE         #define 0x000F = 0x0000000f  intuition/screens.h: *159
SCREEN_DRAG        #define (1<<14) = 0x00004000  intuition/preferences.h: *137
SCRGADGET          #define GTYP_SCRGADGET = 0x00004000
                   intuition/iobsolete.h: *93
SCROLLERIDCMP      #define (IDCMP_GADGETUP | IDCMP_GADGETDOWN | IDCMP_MOUSEMOVE)
                   = 0x00000070
                   libraries/gadtools.h: *78
SCROLLER_KIND      #define 9 = 0x00000009  libraries/gadtools.h: *43
SCSIB_AUTOSENSE    #define 1 = 0x00000001  devices/scsidisk.h: *105
SCSIB_OLDAUTOSENSE #define 2 = 0x00000002  devices/scsidisk.h: *106
SCSIB_READ_WRITE   #define 0 = 0x00000000  devices/scsidisk.h: *98
SCSICmd            structure tag size 0x001e devices/scsidisk.h: *73
SCSIF_AUTOSENSE    #define 2 = 0x00000002  devices/scsidisk.h: *101
SCSIF_NOSENSE      #define 0 = 0x00000000  devices/scsidisk.h: *100
```

Include File Cross Reference Page 120

```
SCSIF_OLDAUTOSENSE #define 6 = 0x00000006  devices/scsidisk.h: *103
SCSIF_READ         #define 1 = 0x00000001  devices/scsidisk.h: *97
SCSIF_WRITE        #define 0 = 0x00000000  devices/scsidisk.h: *96
SDCMD_BREAK        #define (CMD_NONSTD+1) = 0x0000000a  devices/serial.h: *96
SDCMD_QUERY        #define CMD_NONSTD = 0x00000009  devices/serial.h: *95
SDCMD_SETPARAMS    #define (CMD_NONSTD+2) = 0x0000000b  devices/serial.h: *97
SDEPTHIMAGE        #define (0x05L) = 0x00000005  intuition/imageclass.h: *105
SDOWNBACK          #define GTYP_SDOWNBACK = 0x00000070
                   intuition/iobsolete.h: *102
SDOWNBACKGADGET    #define 6 = 0x00000006  intuition/intuitionbase.h: *53
SDRAGGADGET        #define 7 = 0x00000007  intuition/intuitionbase.h: *54
SDRAGGING          #define GTYP_SDRAGGING = 0x00000030
                   intuition/iobsolete.h: *98
SELECTDOWN         #define (IECODE_LBUTTON) = 0x00000068
                   intuition/intuition.h: *1331
SELECTED           #define GFLG_SELECTED = 0x00000080
                   intuition/iobsolete.h: *59
SELECTUP           #define (IECODE_LBUTTON | IECODE_UP_PREFIX) = 0x000000e8
                   intuition/intuition.h: *1330
SERB_7WIRE         #define 2 = 0x00000002  devices/serial.h: *110
SERB_EOFMODE       #define 6 = 0x00000006  devices/serial.h: *102
SERB_PARTY_ODD     #define 1 = 0x00000001  devices/serial.h: *112
SERB_PARTY_ON      #define 0 = 0x00000000  devices/serial.h: *114
SERB_QUEUEDBRK     #define 3 = 0x00000003  devices/serial.h: *108
SERB_RAD_BOOGIE    #define 4 = 0x00000004  devices/serial.h: *106
SERB_SHARED        #define 5 = 0x00000005  devices/serial.h: *104
SERB_XDISABLED     #define 7 = 0x00000007  devices/serial.h: *100
SERF_7WIRE         #define (1<<2) = 0x00000004  devices/serial.h: *111
SERF_EOFMODE       #define (1<<6) = 0x00000040  devices/serial.h: *103
SERF_PARTY_ODD     #define (1<<1) = 0x00000002  devices/serial.h: *113
SERF_PARTY_ON      #define (1<<0) = 0x00000001  devices/serial.h: *115
SERF_QUEUEDBRK     #define (1<<3) = 0x00000008  devices/serial.h: *109
SERF_RAD_BOOGIE    #define (1<<4) = 0x00000010  devices/serial.h: *107
SERF_SHARED        #define (1<<5) = 0x00000020  devices/serial.h: *105
SERF_XDISABLED     #define (1<<7) = 0x00000080  devices/serial.h: *101
SERIALNAME         #define "serial.device"  devices/serial.h: *180
SERIAL_PRINTER     #define 0x01 = 0x00000001  intuition/preferences.h: *142
SER_DEFAULT_CTLCHAR #define 0x11130000 = 0x11130000  devices/serial.h: *28
SEXTB_MARK         #define 0 = 0x00000000  devices/serial.h: *133
SEXTB_MSPON        #define 1 = 0x00000001  devices/serial.h: *130
SEXTF_MARK         #define (1<<0) = 0x00000001  devices/serial.h: *134
SEXTF_MSPON        #define (1<<1) = 0x00000002  devices/serial.h: *132
SGA_BEEP           #define (0x4L) = 0x00000004  intuition/sghooks.h: *116
SGA_END            #define (0x2L) = 0x00000002  intuition/sghooks.h: *115
SGA_NEXTACTIVE     #define (0x20L) = 0x00000020  intuition/sghooks.h: *121
SGA_PREVACTIVE     #define (0x40L) = 0x00000040  intuition/sghooks.h: *122
SGA_REDISPLAY      #define (0x10L) = 0x00000010  intuition/sghooks.h: *118
SGA_REUSE          #define (0x8L) = 0x00000008  intuition/sghooks.h: *117
SGA_USE            #define (0x1L) = 0x00000001  intuition/sghooks.h: *114
SGH_CLICK          #define (2L) = 0x00000002  intuition/sghooks.h: *127
SGH_KEY            #define (1L) = 0x00000001  intuition/sghooks.h: *126
SGM_CONTROL        #define (1L << 5) = 0x00000020  intuition/sghooks.h: *110
SGM_EXITHELP       #define (1L << 7) = 0x00000080  intuition/sghooks.h: *104
SGM_FIXEDFIELD     #define (1L << 1) = 0x00000002  intuition/sghooks.h: *99
SGM_LONGINT        #define (1L << 6) = 0x00000040  intuition/sghooks.h: *111
SGM_NOCHANGE       #define (1L << 3) = 0x00000008  intuition/sghooks.h: *108
SGM_NOFILTER       #define (1L << 2) = 0x00000004  intuition/sghooks.h: *101
SGM_NOWORKB        #define (1L << 4) = 0x00000010  intuition/sghooks.h: *109
SGM_REPLACE        #define (1L << 0) = 0x00000001  intuition/sghooks.h: *94
SGR_BLACK          #define 30 = 0x0000001e  devices/console.h: *43
SGR_BLACKBG        #define 40 = 0x00000028  devices/console.h: *53
SGR_BLUE           #define 34 = 0x00000022  devices/console.h: *47
SGR_BLUEBG         #define 44 = 0x0000002c  devices/console.h: *57
SGR_BOLD           #define 1 = 0x00000001  devices/console.h: *32
SGR_CLR0           #define 30 = 0x0000001e  devices/console.h: *65
SGR_CLR0BG         #define 40 = 0x00000028  devices/console.h: *74
```

```
SGR_CLR1          #define 31 = 0x0000001f  devices/console.h: *66
SGR_CLR1BG        #define 41 = 0x00000029  devices/console.h: *75
SGR_CLR2          #define 32 = 0x00000020  devices/console.h: *67
SGR_CLR2BG        #define 42 = 0x0000002a  devices/console.h: *76
SGR_CLR3          #define 33 = 0x00000021  devices/console.h: *68
SGR_CLR3BG        #define 43 = 0x0000002b  devices/console.h: *77
SGR_CLR4          #define 34 = 0x00000022  devices/console.h: *69
SGR_CLR4BG        #define 44 = 0x0000002c  devices/console.h: *78
SGR_CLR5          #define 35 = 0x00000023  devices/console.h: *70
SGR_CLR5BG        #define 45 = 0x0000002d  devices/console.h: *79
SGR_CLR6          #define 36 = 0x00000024  devices/console.h: *71
SGR_CLR6BG        #define 46 = 0x0000002e  devices/console.h: *80
SGR_CLR7          #define 37 = 0x00000025  devices/console.h: *72
SGR_CLR7BG        #define 47 = 0x0000002f  devices/console.h: *81
SGR_CYAN          #define 36 = 0x00000024  devices/console.h: *49
SGR_CYANBG        #define 46 = 0x0000002e  devices/console.h: *59
SGR_DEFAULT       #define 39 = 0x00000027  devices/console.h: *51
SGR_DEFAULTBG     #define 49 = 0x00000031  devices/console.h: *61
SGR_GREEN         #define 32 = 0x00000020  devices/console.h: *45
SGR_GREENBG       #define 42 = 0x0000002a  devices/console.h: *55
SGR_ITALIC        #define 3 = 0x00000003  devices/console.h: *33
SGR_MAGENTA       #define 35 = 0x00000023  devices/console.h: *48
SGR_MAGENTABG     #define 45 = 0x0000002d  devices/console.h: *58
SGR_NEGATIVE      #define 7 = 0x00000007  devices/console.h: *35
SGR_NORMAL        #define 22 = 0x00000016  devices/console.h: *37
SGR_NOTITALIC     #define 23 = 0x00000017  devices/console.h: *38
SGR_NOTUNDERSCORE #define 24 = 0x00000018  devices/console.h: *39
SGR_POSITIVE      #define 27 = 0x0000001b  devices/console.h: *40
SGR_PRIMARY       #define 0 = 0x00000000  devices/console.h: *31
SGR_RED           #define 31 = 0x0000001f  devices/console.h: *44
SGR_REDBG         #define 41 = 0x00000029  devices/console.h: *54
SGR_UNDERSCORE    #define 4 = 0x00000004  devices/console.h: *34
SGR_WHITE         #define 37 = 0x00000025  devices/console.h: *50
SGR_WHITEBG       #define 47 = 0x0000002f  devices/console.h: *60
SGR_YELLOW        #define 33 = 0x00000021  devices/console.h: *46
SGR_YELLOWBG      #define 43 = 0x0000002b  devices/console.h: *56
SGWork            structure tag size 0x002c intuition/sghooks.h: *33
SG_DEFAULTMAXCHARS #define (128) = 0x00000080  intuition/gadgetclass.h: *147
SHADE_BW          #define 0x00 = 0x00000000  intuition/preferences.h: *180
SHADE_COLOR       #define 0x02 = 0x00000002  intuition/preferences.h: *182
SHADE_GREYSCALE   #define 0x01 = 0x00000001  intuition/preferences.h: *181
SHADOWPEN         #define (0x0004) = 0x00000004  intuition/screens.h: *86
SHAKNUM           macro (1 argument)   intuition/intuition.h: *1281
SHANGHAI          #define 0x0001 = 0x00000001  intuition/screens.h: *397
SHARED_LOCK       #define -2 = 0xfffffffe  dos/dos.h: *49
SHFCprList        pointer to struct cprlist in struct View
          +0x0008 graphics/view.h: *62
SHFlist           pointer to unsigned short int in struct GfxBase
          +0x0036 graphics/gfxbase.h: *33
SHIFTITEM         macro (1 argument)   intuition/intuition.h: *1271
SHIFTMENU         macro (1 argument)   intuition/intuition.h: *1270
SHIFTSUB          macro (1 argument)   intuition/intuition.h: *1272
SHINEPEN          #define (0x0003) = 0x00000003  intuition/screens.h: *85
SHORT             short int  exec/types.h: *56
SHOWTITLE         #define 0x0010 = 0x00000010  intuition/screens.h: *165
SHSHAKE_BITS      #define 0x0F = 0x0000000f  intuition/preferences.h: *223
SHSHAKE_NONE      #define 2 = 0x00000002  intuition/preferences.h: *237
SHSHAKE_RTS       #define 1 = 0x00000001  intuition/preferences.h: *236
SHSHAKE_XON       #define 0 = 0x00000000  intuition/preferences.h: *235
SIGBREAKB_CTRL_C #define 12 = 0x0000000c  dos/dos.h: *204
SIGBREAKB_CTRL_D #define 13 = 0x0000000d  dos/dos.h: *205
SIGBREAKB_CTRL_E #define 14 = 0x0000000e  dos/dos.h: *206
SIGBREAKB_CTRL_F #define 15 = 0x0000000f  dos/dos.h: *207
SIGBREAKF_CTRL_C #define (1<<SIGBREAKB_CTRL_C) = 0x00001000  dos/dos.h: *211
SIGBREAKF_CTRL_D #define (1<<SIGBREAKB_CTRL_D) = 0x00002000  dos/dos.h: *212
SIGBREAKF_CTRL_E #define (1<<SIGBREAKB_CTRL_E) = 0x00004000  dos/dos.h: *213
```

```
SIGBREAKF_CTRL_F #define ((long)1<<SIGBREAKB_CTRL_F) = 0x00008000
                 dos/dos.h: *214
SIGB_ABORT        #define 0 = 0x00000000  exec/tasks.h: *76
SIGB_BLIT         #define 4 = 0x00000004  exec/tasks.h: *78
SIGB_CHILD        #define 1 = 0x00000001  exec/tasks.h: *77
SIGB_DOS          #define 8 = 0x00000008  exec/tasks.h: *81
SIGB_INTUITION    #define 5 = 0x00000005  exec/tasks.h: *80
SIGB_SINGLE       #define 4 = 0x00000004  exec/tasks.h: *79
SIGF_ABORT        #define (1L<<0) = 0x00000001  exec/tasks.h: *83
SIGF_BLIT         #define (1L<<4) = 0x00000010  exec/tasks.h: *85
SIGF_CHILD        #define (1L<<1) = 0x00000002  exec/tasks.h: *84
SIGF_DOS          #define (1L<<8) = 0x00000100  exec/tasks.h: *88
SIGF_INTUITION    #define (1L<<5) = 0x00000020  exec/tasks.h: *87
SIGF_SINGLE       #define (1L<<4) = 0x00000010  exec/tasks.h: *86
SIGN              macro (1 argument)   intuition/intuition.h: *1293
SIGNFLAG          #define 0x40 = 0x00000040  hardware/blit.h: *73
SIH_PRIMASK       #define (0xf0) = 0x000000f0  exec/interrupts.h: *43
SIMPLEREQ         #define 0x0010 = 0x00000010  intuition/intuition.h: *190
SIMPLE_REFRESH    #define WFLG_SIMPLE_REFRESH = 0x00000040
                  intuition/iobsolete.h: *153
SINGLE            #define 0x80 = 0x00000080  intuition/preferences.h: *156
SIX_LPI           #define 0x000 = 0x00000000  intuition/preferences.h: *168
SIZEBBOTTOM       #define WFLG_SIZEBBOTTOM = 0x00000020
                  intuition/iobsolete.h: *150
SIZEBRIGHT        #define WFLG_SIZEBRIGHT = 0x00000010
                  intuition/iobsolete.h: *149
SIZEGADGET        #define 2 = 0x00000002  intuition/intuitionbase.h: *49
SIZEIMAGE         #define (0x02L) = 0x00000002  intuition/imageclass.h: *103
SIZEOF_INSTANCE macro (1 argument)   intuition/classes.h: *52
SIZEVERIFY        #define IDCMP_SIZEVERIFY = 0x00000001
                  intuition/iobsolete.h: *114
SIZING            #define GTYP_SIZING = 0x00000010  intuition/iobsolete.h: *96
SLIDERIDCMP       #define (IDCMP_GADGETUP | IDCMP_GADGETDOWN | IDCMP_MOUSEMOVE)
                  = 0x00000070
                  libraries/gadtools.h: *79
SLIDER_KIND       #define 11 = 0x0000000b  libraries/gadtools.h: *45
SMART_REFRESH     #define WFLG_SMART_REFRESH = 0x00000000
                  intuition/iobsolete.h: *152
SPARITY_BITS      #define 0xF0 = 0x000000f0  intuition/preferences.h: *222
SPARITY_EVEN      #define 1 = 0x00000001  intuition/preferences.h: *229
SPARITY_NONE      #define 0 = 0x00000000  intuition/preferences.h: *228
SPARITY_ODD       #define 2 = 0x00000002  intuition/preferences.h: *230
SPARNUM           macro (1 argument)   intuition/intuition.h: *1280
SPAbs             function returning "LONG"   libraries/mathffp.h: *60
SPAcos            function returning "LONG"   libraries/mathffp.h: *72
SPAdd             function returning "LONG"   libraries/mathffp.h: *67
SPAsin            function returning "LONG"   libraries/mathffp.h: *72
SPAtan            function returning "LONG"   libraries/mathffp.h: *72
SPCeil            function returning "LONG"   libraries/mathffp.h: *62
SPCos             function returning "LONG"   libraries/mathffp.h: *73
SPCosh            function returning "LONG"   libraries/mathffp.h: *74
SPDiv             function returning "LONG"   libraries/mathffp.h: *70
SPECIAL_ASPECT    #define 0x0080 = 0x00000080  devices/printer.h: *182
SPECIAL_BEAMCON   #define ( VARVBLANK | LOLDIS | VARVSYNC | VARBEAM | CSBLANK )
                  = 0x00001a88
                  graphics/monitor.h: *78
SPECIAL_CENTER    #define 0x0040 = 0x00000040  devices/printer.h: *181
SPECIAL_DENSITY1 #define 0x0100 = 0x00000100  devices/printer.h: *183
SPECIAL_DENSITY2 #define 0x0200 = 0x00000200  devices/printer.h: *184
SPECIAL_DENSITY3 #define 0x0300 = 0x00000300  devices/printer.h: *185
SPECIAL_DENSITY4 #define 0x0400 = 0x00000400  devices/printer.h: *186
SPECIAL_DENSITY5 #define 0x0500 = 0x00000500  devices/printer.h: *187
SPECIAL_DENSITY6 #define 0x0600 = 0x00000600  devices/printer.h: *188
SPECIAL_DENSITY7 #define 0x0700 = 0x00000700  devices/printer.h: *189
SPECIAL_DENSITYMASK #define 0x0700 = 0x00000700  devices/printer.h: *221
SPECIAL_DIMENSIONSMASK #define (SPECIAL_MILCOLS|SPECIAL_MILROWS|
```

Include File Cross Reference Page 123

```
                                SPECIAL_FULLCOLS|SPECIAL_FULLROWS |
                                SPECIAL_FRACCOLS|SPECIAL_FRACROWS|
                                SPECIAL_ASPECT) = 0x000000bf
                  devices/printer.h: *224
SPECIAL_FRACCOLS #define 0x0010 = 0x00000010  devices/printer.h: *179
SPECIAL_FRACROWS #define 0x0020 = 0x00000020  devices/printer.h: *180
SPECIAL_FULLCOLS #define 0x0004 = 0x00000004  devices/printer.h: *177
SPECIAL_FULLROWS #define 0x0008 = 0x00000008  devices/printer.h: *178
SPECIAL_MILCOLS  #define 0x0001 = 0x00000001  devices/printer.h: *175
SPECIAL_MILROWS  #define 0x0002 = 0x00000002  devices/printer.h: *176
SPECIAL_MONITOR_TYPE #define 3 = 0x00000003  graphics/gfxnodes.h: *35
SPECIAL_NOFORMFEED #define 0x0800 = 0x00000800  devices/printer.h: *190
SPECIAL_NOPRINT  #define 0x2000 = 0x00002000  devices/printer.h: *201
SPECIAL_TRUSTME  #define 0x1000 = 0x00001000  devices/printer.h: *191
SPExp             function returning "LONG"   libraries/mathffp.h: *75
SPFieee           function returning "LONG"   libraries/mathffp.h: *76
SPFloor           function returning "LONG"   libraries/mathffp.h: *61
SPFlt             function returning "LONG"   libraries/mathffp.h: *57
SPLog             function returning "LONG"   libraries/mathffp.h: *75
SPLog10           function returning "LONG"   libraries/mathffp.h: *75
SPMul             function returning "LONG"   libraries/mathffp.h: *69
SPNeg             function returning "LONG"   libraries/mathffp.h: *66
SPPow             function returning "LONG"   libraries/mathffp.h: *75
SPRITES           #define 0x4000 = 0x00004000  graphics/view.h: *100
SPRITE_ATTACHED   #define 0x80 = 0x00000080  graphics/sprite.h: *19
SPSin             function returning "LONG"   libraries/mathffp.h: *73
SPSincos          function returning "LONG"   libraries/mathffp.h: *73
SPSinh            function returning "LONG"   libraries/mathffp.h: *74
SPSqrt            function returning "LONG"   libraries/mathffp.h: *76
SPSub             function returning "LONG"   libraries/mathffp.h: *68
SPTan             function returning "LONG"   libraries/mathffp.h: *73
SPTanh            function returning "LONG"   libraries/mathffp.h: *74
SRBNUM            macro (1 argument)   intuition/intuition.h: *1277
SRCA              #define 0x800 = 0x00000800  hardware/blit.h: *61
SRCB              #define 0x400 = 0x00000400  hardware/blit.h: *60
SRCC              #define 0x200 = 0x00000200  hardware/blit.h: *59
SREAD_BITS        #define 0xF0 = 0x000000f0  intuition/preferences.h: *216
SSBNUM            macro (1 argument)   intuition/intuition.h: *1279
SSTOP_BITS        #define 0xF0 = 0x000000f0  intuition/preferences.h: *219
SS_GRAPHICS       #define 0x02 = 0x00000002  graphics/gfxnodes.h: *31
STANDARD_COLORCLOCKS #define 226 = 0x000000e2  graphics/monitor.h: *72
STANDARD_DENISE_MAX #define 455 = 0x000001c7  graphics/monitor.h: *73
STANDARD_DENISE_MIN #define 93 = 0x0000005d  graphics/monitor.h: *74
STANDARD_HBSTOP   #define 0x2C = 0x0000002c  graphics/monitor.h: *87
STANDARD_HBSTRT   #define 0x06 = 0x00000006  graphics/monitor.h: *84
STANDARD_HSSTOP   #define 0x1C = 0x0000001c  graphics/monitor.h: *86
STANDARD_HSSTRT   #define 0x0B = 0x0000000b  graphics/monitor.h: *85
STANDARD_MONITOR_MASK #define ( REQUEST_NTSC | REQUEST_PAL ) = 0x00000003
                  graphics/monitor.h: *68
STANDARD_NTSC_BEAMCON #define ( 0x0000 ) = 0x00000000  graphics/monitor.h: *75
STANDARD_NTSC_ROWS #define 262 = 0x00000106  graphics/monitor.h: *70
STANDARD_PAL_BEAMCON #define ( DISPLAYPAL ) = 0x00000020
                  graphics/monitor.h: *76
STANDARD_PAL_ROWS #define 312 = 0x00000138  graphics/monitor.h: *71
STANDARD_VBSTOP   #define 0x1066 = 0x00001066  graphics/monitor.h: *91
STANDARD_VBSTRT   #define 0x0122 = 0x00000122  graphics/monitor.h: *88
STANDARD_VIEW_X   #define 0x81 = 0x00000081  graphics/monitor.h: *82
STANDARD_VIEW_Y   #define 0x2C = 0x0000002c  graphics/monitor.h: *83
STANDARD_VSSTOP   #define 0x03AA = 0x000003aa  graphics/monitor.h: *90
STANDARD_VSSTRT   #define 0x02A6 = 0x000002a6  graphics/monitor.h: *89
STANDARD_XOFFSET #define 9 = 0x00000009  graphics/monitor.h: *58
STANDARD_YOFFSET #define 0 = 0x00000000  graphics/monitor.h: *59
STATIC            #define static  exec/types.h: *22
STDSCREENHEIGHT   #define -1 = 0xffffffff  intuition/screens.h: *184
STDSCREENWIDTH    #define -1 = 0xffffffff  intuition/screens.h: *185
STRGADGET         #define GTYP_STRGADGET = 0x00000004
```

Include File Cross Reference Page 124

```
                  intuition/iobsolete.h: *107
STRGCLASS         #define "strgclass"  intuition/classusr.h: *49
STRINGA_ActivePens #define (STRINGA_Dummy + 0x000A) = 0x8003200a
                  intuition/gadgetclass.h: *127
STRINGA_AltKeyMap #define (STRINGA_Dummy + 0x0007) = 0x80032007
                  intuition/gadgetclass.h: *124
STRINGA_Buffer    #define (STRINGA_Dummy + 0x0002) = 0x80032002
                  intuition/gadgetclass.h: *119
STRINGA_BufferPos #define (STRINGA_Dummy + 0x0005) = 0x80032005
                  intuition/gadgetclass.h: *122
STRINGA_DispPos   #define (STRINGA_Dummy + 0x0006) = 0x80032006
                  intuition/gadgetclass.h: *123
STRINGA_Dummy     #define (TAG_USER +0x32000) = 0x80032000
                  intuition/gadgetclass.h: *117
STRINGA_EditHook #define (STRINGA_Dummy + 0x000B) = 0x8003200b
                  intuition/gadgetclass.h: *128
STRINGA_EditModes #define (STRINGA_Dummy + 0x000C) = 0x8003200c
                  intuition/gadgetclass.h: *129
STRINGA_ExitHelp #define (STRINGA_Dummy + 0x0013) = 0x80032013
                  intuition/gadgetclass.h: *141
STRINGA_FixedFieldMode #define (STRINGA_Dummy + 0x000E) = 0x8003200e
                  intuition/gadgetclass.h: *133
STRINGA_Font      #define (STRINGA_Dummy + 0x0008) = 0x80032008
                  intuition/gadgetclass.h: *125
STRINGA_Justification #define (STRINGA_Dummy + 0x0010) = 0x80032010
                  intuition/gadgetclass.h: *136
STRINGA_LongVal   #define (STRINGA_Dummy + 0x0011) = 0x80032011
                  intuition/gadgetclass.h: *138
STRINGA_MaxChars #define (STRINGA_Dummy + 0x0001) = 0x80032001
                  intuition/gadgetclass.h: *118
STRINGA_NoFilterMode #define (STRINGA_Dummy + 0x000F) = 0x8003200f
                  intuition/gadgetclass.h: *134
STRINGA_Pens      #define (STRINGA_Dummy + 0x0009) = 0x80032009
                  intuition/gadgetclass.h: *126
STRINGA_ReplaceMode #define (STRINGA_Dummy + 0x000D) = 0x8003200d
                  intuition/gadgetclass.h: *132
STRINGA_TextVal   #define (STRINGA_Dummy + 0x0012) = 0x80032012
                  intuition/gadgetclass.h: *139
STRINGA_UndoBuffer #define (STRINGA_Dummy + 0x0003) = 0x80032003
                  intuition/gadgetclass.h: *120
STRINGA_WorkBuffer #define (STRINGA_Dummy + 0x0004) = 0x80032004
                  intuition/gadgetclass.h: *121
STRINGCENTER      #define GACT_STRINGCENTER = 0x00000200
                  intuition/iobsolete.h: *81
STRINGEXTEND      #define GACT_STRINGEXTEND = 0x00002000
                  intuition/iobsolete.h: *85
STRINGIDCMP       #define (IDCMP_GADGETUP) = 0x00000040
                  libraries/gadtools.h: *80
STRINGLEFT        #define GACT_STRINGLEFT = 0x00000000
                  intuition/iobsolete.h: *80
STRINGRIGHT       #define GACT_STRINGRIGHT = 0x00000400
                  intuition/iobsolete.h: *82
STRING_KIND       #define 12 = 0x0000000c  libraries/gadtools.h: *46
STRPTR            pointer to unsigned char  exec/types.h: *52
ST_FILE           #define -3 = 0xfffffffd  dos/dosextens.h: *477
ST_LINKDIR        #define 4 = 0x00000004  dos/dosextens.h: *476
ST_LINKFILE       #define -4 = 0xfffffffc  dos/dosextens.h: *478
ST_ROOT           #define 1 = 0x00000001  dos/dosextens.h: *473
ST_SOFTLINK       #define 3 = 0x00000003  dos/dosextens.h: *475
ST_USERDIR        #define 2 = 0x00000002  dos/dosextens.h: *474
SUBNUM            macro (1 argument)   intuition/intuition.h: *1268
SUD               #define 0x10 = 0x00000010  hardware/blit.h: *76
SUL               #define 0x8 = 0x00000008  hardware/blit.h: *77
SUPERDPF2_KEY     #define 0x00008460 = 0x00008460  graphics/displayinfo.h: *171
SUPERDPF_KEY      #define 0x00008420 = 0x00008420  graphics/displayinfo.h: *165
SUPERHIRES        #define 0x0020 = 0x00000020  graphics/view.h: *92
```

Include File Cross Reference Page 125

```
SUPERLACEDPF2_KEY #define 0x00008464 = 0x00008464
                  graphics/displayinfo.h: *174
SUPERLACEDPF_KEY #define 0x00008424 = 0x00008424  graphics/displayinfo.h: *168
SUPERLACE_KEY    #define 0x00008024 = 0x00008024  graphics/displayinfo.h: *161
SUPER_BITMAP     #define WFLG_SUPER_BITMAP = 0x00000080
                 intuition/iobsolete.h: *154
SUPER_KEY        #define 0x00008020 = 0x00008020  graphics/displayinfo.h: *157
SUPFRONT         #define GTYP_SUPFRONT = 0x00000050
                 intuition/iobsolete.h: *100
SUPFRONTGADGET   #define 5 = 0x00000005  intuition/intuitionbase.h: *52
SUSERFLAGS       #define 0x00FF = 0x000000ff  graphics/gels.h: *21
SWBNUM           macro (1 argument)   intuition/intuition.h: *1278
SWRITE_BITS      #define 0x0F = 0x0000000f  intuition/preferences.h: *217
SYSGADGET        #define GTYP_SYSGADGET = 0x00008000
                 intuition/iobsolete.h: *92
SYSIA_Depth      #define (IA_Dummy + 0x0C) = 0x8002000c
                 intuition/imageclass.h: *74
SYSIA_DrawInfo   #define (IA_Dummy + 0x18) = 0x80020018
                 intuition/imageclass.h: *80
SYSIA_Pens       #define IA_Pens = 0x8002000e  intuition/imageclass.h: *84
SYSIA_Size       #define (IA_Dummy + 0x0B) = 0x8002000b
                 intuition/imageclass.h: *72
SYSIA_Which      #define (IA_Dummy + 0x0D) = 0x8002000d
                 intuition/imageclass.h: *78
SYSICLASS        #define "sysiclass"  intuition/classusr.h: *45
SYSISIZE_HIRES   #define (2) = 0x00000002  intuition/imageclass.h: *94
SYSISIZE_LOWRES  #define (1) = 0x00000001  intuition/imageclass.h: *93
SYSISIZE_MEDRES  #define (0) = 0x00000000  intuition/imageclass.h: *92
SYSREQUEST       #define 0x4000 = 0x00004000  intuition/intuition.h: *205
SYS_Asynch       #define (SYS_Dummy + 3) = 0x80000023  dos/dostags.h: *29
SYS_CustomShell  #define (SYS_Dummy + 5) = 0x80000025  dos/dostags.h: *33
SYS_Dummy        #define (TAG_USER + 32) = 0x80000020  dos/dostags.h: *24
SYS_Input        #define (SYS_Dummy + 1) = 0x80000021  dos/dostags.h: *25
SYS_Output       #define (SYS_Dummy + 2) = 0x80000022  dos/dostags.h: *27
SYS_UserShell    #define (SYS_Dummy + 4) = 0x80000024  dos/dostags.h: *31
SatisfyMsg       structure tag size 0x001a devices/clipboard.h: *59
SaveBuffer       pointer to short int in struct Bob
         +0x0002 graphics/gels.h: *145
SaveColor0       unsigned short int in struct Screen
         +0x014c intuition/screens.h: *142
Screen           structure tag size 0x015a devices/inputevent.h: *89
                 intuition/intuition.h: 819, 1005, 1057
                 intuition/screens.h: 98, 100, 384
                 intuition/cghooks.h: 29
                 intuition/intuitionbase.h: 75, 80
Screen           pointer to struct Screen in struct NewWindow
         +0x001e intuition/intuition.h: *1005
Screen           pointer to struct Screen in struct ExtNewWindow
         +0x001e intuition/intuition.h: *1057
ScreenTitle      pointer to unsigned char in struct Window
         +0x0068 intuition/intuition.h: *867
Scroll_X         short int in struct Layer +0x002c graphics/clip.h: *47
Scroll_Y         short int in struct Layer +0x002e graphics/clip.h: *47
Seconds          unsigned long int in struct IntuiMessage
         +0x0024 intuition/intuition.h: *708
Seconds          unsigned long int in struct IntuitionBase
         +0x0048 intuition/intuitionbase.h: *86
Segment          structure tag size 0x0010 dos/dosextens.h: *288
SelectFill       pointer to void in struct MenuItem
         +0x0016 intuition/intuition.h: *104
SelectRender     pointer to void in struct Gadget
         +0x0016 intuition/intuition.h: *238
Semaphore        structure tag size 0x0024 exec/semaphores.h: *54
SemaphoreList    struct List(size 0x000e bytes) in struct ExecBase
         +0x0214 exec/execbase.h: *114
SemaphoreRequest structure tag size 0x000c exec/semaphores.h: *35, 45
```

Include File Cross Reference Page 126

```
SerErr_BaudMismatch #define 2 = 0x00000002  devices/serial.h: *138
SerErr_BufErr    #define 4 = 0x00000004  devices/serial.h: *139
SerErr_BufOverflow #define 12 = 0x0000000c  devices/serial.h: *144
SerErr_DetectedBreak #define 15 = 0x0000000f  devices/serial.h: *146
SerErr_DevBusy   #define 1 = 0x00000001  devices/serial.h: *137
SerErr_InvParam  #define 5 = 0x00000005  devices/serial.h: *140
SerErr_LineErr   #define 6 = 0x00000006  devices/serial.h: *141
SerErr_NoDSR     #define 13 = 0x0000000d  devices/serial.h: *145
SerErr_ParityErr #define 9 = 0x00000009  devices/serial.h: *142
SerErr_TimerErr  #define 11 = 0x0000000b  devices/serial.h: *143
SerParShk        unsigned char in struct Preferences
         +0x00b8 intuition/preferences.h: *108
SerRWBits        unsigned char in struct Preferences
         +0x00b6 intuition/preferences.h: *104
SerStopBuf       unsigned char in struct Preferences
         +0x00b7 intuition/preferences.h: *106
SetAfPt          macro (3 arguments)   graphics/gfxmacros.h: *34
SetDrPt          macro (2 arguments)   graphics/gfxmacros.h: *32
SetOPen          macro (2 arguments)   graphics/gfxmacros.h: *31
SetWrMsk         macro (2 arguments)   graphics/gfxmacros.h: *33
SignalSemaphore structure tag size 0x002e exec/semaphores.h: *41
                 dos/dosextens.h: 281, 282
                 graphics/clip.h: 52
                 graphics/monitor.h: 51
                 graphics/layers.h: 40
                 graphics/gfxbase.h: 63, 88, 90
SimpleSprite     structure tag size 0x000c graphics/gfxbase.h: *55
                 graphics/sprite.h: 21
SimpleSprites    pointer to pointer to struct SimpleSprite in struct GfxBase
         +0x00d0 graphics/gfxbase.h: *55
Size             long int in struct TmpRas +0x0004 graphics/rastport.h: *37
SkipID           unsigned long int in struct QueryHeader
         +0x0008 graphics/displayinfo.h: *46
SoftIntList      structure tag size 0x0010 exec/interrupts.h: *38
                 exec/execbase.h: 95
SoftInts         array [5] of struct SoftIntList(size 0x0010 bytes) in struct
                 ExecBase
         +0x01b2 exec/execbase.h: *95
SoftVer          unsigned short int in struct ExecBase
         +0x0022 exec/execbase.h: *41
SpecialInfo      pointer to void in struct Gadget
         +0x0022 intuition/intuition.h: *258
SpecialLink      pointer to struct IntuiMessage in struct IntuiMessage
         +0x0030 intuition/intuition.h: *716
SpecialMonitor   structure tag size 0x003a graphics/monitor.h: *38, 142
SprColors        pointer to short int in struct VSprite
         +0x0030 graphics/gels.h: *114
SprIns           pointer to struct CopList in struct ViewPort
         +0x000c graphics/view.h: *47
SpriteDef        structure tag (size 0x0008 bytes)in struct Custom
                 hardware/custom.h: *116
SpritePriorities unsigned char in struct ViewPort +0x0022 graphics/view.h: *53
SpriteReserved   unsigned char in struct GfxBase
         +0x00a6 graphics/gfxbase.h: *44
SpriteResolution struct tPoint(size 0x0004 bytes) in struct DisplayInfo
         +0x0020 graphics/displayinfo.h: *59
SrcNode          structure tag size 0x0010 rexx/storage.h: *227, 228, 229
StandardPacket   structure tag size 0x0044 dos/dosextens.h: *141
StdOScan         struct Rectangle(size 0x0008 bytes) in struct DimensionInfo
         +0x003a graphics/displayinfo.h: *104
StoredProperty   structure tag size 0x0008 libraries/iffparse.h: *95
StringExtend     structure tag size 0x0024 intuition/intuition.h: *546
                 intuition/sghooks.h: 19
StringInfo       structure tag size 0x0024 intuition/intuition.h: *524
                 intuition/sghooks.h: 36
StringInfo       pointer to struct StringInfo in struct SGWork
```

```
          +0x0004 intuition/sghooks.h: *36
StructID          unsigned long int in struct QueryHeader
          +0x0000 graphics/displayinfo.h: *44
SubItem           pointer to struct MenuItem in struct MenuItem
          +0x001c intuition/intuition.h: *108
SuperBitMap       pointer to struct BitMap in struct Layer
          +0x0020 graphics/clip.h: *43
SuperClipRect     pointer to struct ClipRect in struct Layer
          +0x0024 graphics/clip.h: *44
SuperSaveClipRects pointer to struct ClipRect in struct Layer
          +0x003c graphics/clip.h: *49
SysFlags          unsigned short int in struct ExecBase
          +0x0124 exec/execbase.h: *69
SysStkLower       pointer to void in struct ExecBase
          +0x003a exec/execbase.h: *48
SysStkUpper       pointer to void in struct ExecBase
          +0x0036 exec/execbase.h: *47
tPoint            structure tag size 0x0004 graphics/gfx.h: *41
ta_Flags          unsigned char in struct TextAttr +0x0007 graphics/text.h: *71
ta_Name           pointer to unsigned char in struct TextAttr
          +0x0000 graphics/text.h: *68
ta_Style          unsigned char in struct TextAttr +0x0006 graphics/text.h: *70
ta_YSize          unsigned short int in struct TextAttr
          +0x0004 graphics/text.h: *69
taf_Attr          struct TTextAttr(size 0x000c bytes) in struct TAvailFonts
          +0x0002 libraries/diskfont.h: *102
taf_Type          unsigned short int in struct TAvailFonts
          +0x0000 libraries/diskfont.h: *101
tan               #define IEEEDPTan  libraries/mathffp.h: *39
                  libraries/mathieeedp.h: *39
tanh              #define IEEEDPTanh  libraries/mathffp.h: *53
                  libraries/mathieeedp.h: *53
tc_ExceptCode     pointer to void in struct Task +0x002a exec/tasks.h: *39
tc_ExceptData     pointer to void in struct Task +0x0026 exec/tasks.h: *38
tc_Flags          unsigned char in struct Task +0x000e exec/tasks.h: *28
tc_IDNestCnt      char in struct Task +0x0010 exec/tasks.h: *30
tc_Launch         pointer to function returning void in struct Task
          +0x0046 exec/tasks.h: *46
tc_MemEntry       struct List(size 0x000e bytes) in struct Task
          +0x004a exec/tasks.h: *47
tc_Node           struct Node(size 0x000e bytes) in struct Task
          +0x0000 exec/tasks.h: *27
tc_SPLower        pointer to void in struct Task +0x003a exec/tasks.h: *43
tc_SPReg          pointer to void in struct Task +0x0036 exec/tasks.h: *42
tc_SPUpper        pointer to void in struct Task +0x003e exec/tasks.h: *44
tc_SigAlloc       unsigned long int in struct Task +0x0012 exec/tasks.h: *32
tc_SigExcept      unsigned long int in struct Task +0x001e exec/tasks.h: *35
tc_SigRecvd       unsigned long int in struct Task +0x001a exec/tasks.h: *34
tc_SigWait        unsigned long int in struct Task +0x0016 exec/tasks.h: *33
tc_State          unsigned char in struct Task +0x000f exec/tasks.h: *29
tc_Switch         pointer to function returning void in struct Task
          +0x0042 exec/tasks.h: *45
tc_TDNestCnt      char in struct Task +0x0011 exec/tasks.h: *31
tc_TrapAble       unsigned short int in struct Task +0x0024 exec/tasks.h: *37
tc_TrapAlloc      unsigned short int in struct Task +0x0022 exec/tasks.h: *36
tc_TrapCode       pointer to void in struct Task +0x0032 exec/tasks.h: *41
tc_TrapData       pointer to void in struct Task +0x002e exec/tasks.h: *40
tc_UserData       pointer to void in struct Task +0x0058 exec/tasks.h: *48
tdu_CalibrateDelay unsigned long int in struct TDU_PublicUnit
          +0x0038 devices/trackdisk.h: *251
tdu_Comp01Track unsigned short int in struct TDU_PublicUnit
          +0x0026 devices/trackdisk.h: *242
tdu_Comp10Track unsigned short int in struct TDU_PublicUnit
          +0x0028 devices/trackdisk.h: *243
tdu_Comp11Track unsigned short int in struct TDU_PublicUnit
          +0x002a devices/trackdisk.h: *244
```

```
tdu_Counter       unsigned long int in struct TDU_PublicUnit
          +0x003c devices/trackdisk.h: *253
tdu_CurrTrk       unsigned short int in struct TDU_PublicUnit
          +0x0036 devices/trackdisk.h: *249
tdu_PubFlags      unsigned char in struct TDU_PublicUnit
          +0x0035 devices/trackdisk.h: *248
tdu_RetryCnt      unsigned char in struct TDU_PublicUnit
          +0x0034 devices/trackdisk.h: *247
tdu_SettleDelay unsigned long int in struct TDU_PublicUnit
          +0x0030 devices/trackdisk.h: *246
tdu_StepDelay     unsigned long int in struct TDU_PublicUnit
          +0x002c devices/trackdisk.h: *245
tdu_Unit          struct Unit(size 0x0026 bytes) in struct TDU_PublicUnit
          +0x0000 devices/trackdisk.h: *241
te_Extent         struct Rectangle(size 0x0008 bytes) in struct TextExtent
          +0x0004 graphics/text.h: *169
te_Height         unsigned short int in struct TextExtent
          +0x0002 graphics/text.h: *168
te_Width          unsigned short int in struct TextExtent
          +0x0000 graphics/text.h: *167
textPen           #define TEXTPEN = 0x00000002  intuition/iobsolete.h: *263
tf_Accessors      unsigned short int in struct TextFont
          +0x001e graphics/text.h: *101
tf_Baseline       unsigned short int in struct TextFont
          +0x001a graphics/text.h: *98
tf_BoldSmear      unsigned short int in struct TextFont
          +0x001c graphics/text.h: *99
tf_CharData       pointer to void in struct TextFont
          +0x0022 graphics/text.h: *105
tf_CharKern       pointer to void in struct TextFont
          +0x0030 graphics/text.h: *111
tf_CharLoc        pointer to void in struct TextFont
          +0x0028 graphics/text.h: *108
tf_CharSpace      pointer to void in struct TextFont
          +0x002c graphics/text.h: *110
tf_Extension      #define tf_Message.mn_ReplyPort  graphics/text.h: *115
tf_Flags          unsigned char in struct TextFont +0x0017 graphics/text.h: *96
tf_HiChar         unsigned char in struct TextFont +0x0021 graphics/text.h: *104
tf_LoChar         unsigned char in struct TextFont +0x0020 graphics/text.h: *103
tf_Message        struct Message(size 0x0014 bytes) in struct TextFont
          +0x0000 graphics/text.h: *92
tf_Modulo         unsigned short int in struct TextFont
          +0x0026 graphics/text.h: *107
tf_Style          unsigned char in struct TextFont +0x0016 graphics/text.h: *95
tf_XSize          unsigned short int in struct TextFont
          +0x0018 graphics/text.h: *97
tf_YSize          unsigned short int in struct TextFont
          +0x0014 graphics/text.h: *94
tfc_FileName      array [254] of char in struct TFontContents
          +0x0000 libraries/diskfont.h: *38
tfc_Flags         unsigned char in struct TFontContents
          +0x0103 libraries/diskfont.h: *47
tfc_Style         unsigned char in struct TFontContents
          +0x0102 libraries/diskfont.h: *46
tfc_TagCount      unsigned short int in struct TFontContents
          +0x00fe libraries/diskfont.h: *39
tfc_YSize         unsigned short int in struct TFontContents
          +0x0100 libraries/diskfont.h: *45
tfe_BackPtr       pointer to struct TextFont in struct TextFontExtension
          +0x0004 graphics/text.h: *125
tfe_Flags0        unsigned char in struct TextFontExtension
          +0x0002 graphics/text.h: *123
tfe_Flags1        unsigned char in struct TextFontExtension
          +0x0003 graphics/text.h: *124
tfe_MatchWord     unsigned short int in struct TextFontExtension
          +0x0000 graphics/text.h: *122
```

Include File Cross Reference Page 129

```
tfe_OFontPatchK pointer to unsigned short int in struct TextFontExtension
           +0x0014 graphics/text.h: *129
tfe_OFontPatchS pointer to unsigned short int in struct TextFontExtension
           +0x0010 graphics/text.h: *128
tfe_OrigReplyPort pointer to struct MsgPort in struct TextFontExtension
           +0x0008 graphics/text.h: *126
tfe_Tags           pointer to struct TagItem in struct TextFontExtension
           +0x000c graphics/text.h: *127
ti_Data            unsigned long int in struct TagItem
           +0x0004 utility/tagitem.h: *34
ti_Tag             unsigned long int in struct TagItem
           +0x0000 utility/tagitem.h: *33
timerequest        structure tag size 0x0028 devices/timer.h: *37
                   dos/dosextens.h: 236
                   devices/prtbase.h: 91
timeval            structure tag size 0x0008 devices/timer.h: *27, 39
                   devices/inputevent.h: 218
                   intuition/preferences.h: 56, 57, 58
timsrv             struct Interrupt(size 0x0016 bytes) in struct GfxBase
           +0x0060 graphics/gfxbase.h: *36
top_layer          pointer to struct Layer in struct Layer_Info
           +0x0000 graphics/layers.h: *36
topmost            short int in struct GelsInfo +0x001a graphics/rastport.h: *52
total_colorclocks unsigned short int in struct MonitorSpec
           +0x0024 graphics/monitor.h: *34
total_rows         unsigned short int in struct MonitorSpec
           +0x0022 graphics/monitor.h: *33
tr_node            struct IORequest(size 0x0020 bytes) in struct timerequest
           +0x0000 devices/timer.h: *38
tr_time            struct timeval(size 0x0008 bytes) in struct timerequest
           +0x0020 devices/timer.h: *39
trunc              macro (1 argument)   libraries/mathffp.h: *31
                   libraries/mathieeedp.h: *31
tta_Flags          unsigned char in struct TTextAttr +0x0007 graphics/text.h: *78
tta_Name           pointer to unsigned char in struct TTextAttr
           +0x0000 graphics/text.h: *75
tta_Style          unsigned char in struct TTextAttr +0x0006 graphics/text.h: *77
tta_Tags           pointer to struct TagItem in struct TTextAttr
           +0x0008 graphics/text.h: *79
tta_YSize          unsigned short int in struct TTextAttr
           +0x0004 graphics/text.h: *76
tv_micro           unsigned long int in struct timeval
           +0x0004 devices/timer.h: *29
tv_secs            unsigned long int in struct timeval
           +0x0000 devices/timer.h: *28
TAGFILTER_AND      #define 0 = 0x00000000  utility/tagitem.h: *54
TAGFILTER_NOT      #define 1 = 0x00000001  utility/tagitem.h: *55
TAG_DONE           #define (0L) = 0x00000000  utility/tagitem.h: *38
TAG_END            #define TAG_DONE = 0x00000000  utility/tagitem.h: *39
TAG_IGNORE         #define (1L) = 0x00000001  utility/tagitem.h: *40
TAG_MORE           #define (2L) = 0x00000002  utility/tagitem.h: *43
TAG_SKIP           #define (3L) = 0x00000003  utility/tagitem.h: *44
TAG_USER           #define (1L<<31) = 0x80000000  utility/tagitem.h: *47
TA_DeviceDPI       #define (1|TAG_USER) = 0x80000001  graphics/text.h: *84
TAvailFonts        structure tag size 0x000e libraries/diskfont.h: *100
TBC_HCLRTAB        #define 0 = 0x00000000  devices/console.h: *94
TBC_HCLRTABSALL    #define 3 = 0x00000003  devices/console.h: *95
TB_ETASK           #define 3 = 0x00000003  exec/tasks.h: *53
TB_EXCEPT          #define 5 = 0x00000005  exec/tasks.h: *55
TB_LAUNCH          #define 7 = 0x00000007  exec/tasks.h: *57
TB_PROCTIME        #define 0 = 0x00000000  exec/tasks.h: *52
TB_STACKCHK        #define 4 = 0x00000004  exec/tasks.h: *54
TB_SWITCH          #define 6 = 0x00000006  exec/tasks.h: *56
TDB_ALLOW_NON_3_5 #define 0 = 0x00000000  devices/trackdisk.h: *193
TDERR_BadDriveType #define 33 = 0x00000021  devices/trackdisk.h: *228
TDERR_BadHdrSum    #define 24 = 0x00000018  devices/trackdisk.h: *219
```

Include File Cross Reference Page 130

```
TDERR_BadSecHdr    #define 27 = 0x0000001b  devices/trackdisk.h: *222
TDERR_BadSecID     #define 23 = 0x00000017  devices/trackdisk.h: *218
TDERR_BadSecPreamble #define 22 = 0x00000016  devices/trackdisk.h: *217
TDERR_BadSecSum    #define 25 = 0x00000019  devices/trackdisk.h: *220
TDERR_BadUnitNum   #define 32 = 0x00000020  devices/trackdisk.h: *227
TDERR_DiskChanged #define 29 = 0x0000001d  devices/trackdisk.h: *224
TDERR_DriveInUse   #define 34 = 0x00000022  devices/trackdisk.h: *229
TDERR_NoMem        #define 31 = 0x0000001f  devices/trackdisk.h: *226
TDERR_NoSecHdr     #define 21 = 0x00000015  devices/trackdisk.h: *216
TDERR_NotSpecified #define 20 = 0x00000014  devices/trackdisk.h: *215
TDERR_PostReset    #define 35 = 0x00000023  devices/trackdisk.h: *230
TDERR_SeekError    #define 30 = 0x0000001e  devices/trackdisk.h: *225
TDERR_TooFewSecs   #define 26 = 0x0000001a  devices/trackdisk.h: *221
TDERR_WriteProt    #define 28 = 0x0000001c  devices/trackdisk.h: *223
TDF_ALLOW_NON_3_5 #define (1<<0) = 0x00000001  devices/trackdisk.h: *194
TDF_EXTCOM         #define (1<<15) = 0x00008000  devices/trackdisk.h: *76
TDNestCnt          char in struct ExecBase +0x0127 exec/execbase.h: *71
TDPB_NOCLICK       #define 0 = 0x00000000  devices/trackdisk.h: *258
TDPF_NOCLICK       #define (1L << 0) = 0x00000001  devices/trackdisk.h: *259
TDU_PublicUnit     structure tag size 0x0040 devices/trackdisk.h: *240
TD_ADDCHANGEINT    #define (CMD_NONSTD+11) = 0x00000014
                   devices/trackdisk.h: *90
TD_CHANGENUM       #define (CMD_NONSTD+4) = 0x0000000d  devices/trackdisk.h: *83
TD_CHANGESTATE     #define (CMD_NONSTD+5) = 0x0000000e  devices/trackdisk.h: *84
TD_EJECT           #define (CMD_NONSTD+14) = 0x00000017
                   devices/trackdisk.h: *93
TD_FORMAT          #define (CMD_NONSTD+2) = 0x0000000b  devices/trackdisk.h: *81
TD_GETDRIVETYPE    #define (CMD_NONSTD+9) = 0x00000012  devices/trackdisk.h: *88
TD_GETGEOMETRY     #define (CMD_NONSTD+13) = 0x00000016
                   devices/trackdisk.h: *92
TD_GETNUMTRACKS    #define (CMD_NONSTD+10) = 0x00000013
                   devices/trackdisk.h: *89
TD_LABELSIZE       #define 16 = 0x00000010  devices/trackdisk.h: *185
TD_LASTCOMM        #define (CMD_NONSTD+15) = 0x00000018
                   devices/trackdisk.h: *94
TD_MOTOR           #define (CMD_NONSTD+0) = 0x00000009  devices/trackdisk.h: *79
TD_NAME            #define "trackdisk.device"  devices/trackdisk.h: *74
TD_PROTSTATUS      #define (CMD_NONSTD+6) = 0x0000000f  devices/trackdisk.h: *85
TD_RAWREAD         #define (CMD_NONSTD+7) = 0x00000010  devices/trackdisk.h: *86
TD_RAWWRITE        #define (CMD_NONSTD+8) = 0x00000011  devices/trackdisk.h: *87
TD_REMCHANGEINT    #define (CMD_NONSTD+12) = 0x00000015
                   devices/trackdisk.h: *91
TD_REMOVE          #define (CMD_NONSTD+3) = 0x0000000c  devices/trackdisk.h: *82
TD_SECSHIFT        #define 9 = 0x00000009  devices/trackdisk.h: *53
TD_SECTOR          #define 512 = 0x00000200  devices/trackdisk.h: *52
TD_SEEK            #define (CMD_NONSTD+1) = 0x0000000a  devices/trackdisk.h: *80
TEOB_NOREMFONT     #define 0 = 0x00000000  graphics/text.h: *118
TEOF_NOREMFONT     #define 0x01 = 0x00000001  graphics/text.h: *119
TEXT               typedef unsigned char  exec/types.h: *67
TEXTIDCMP          #define (NULL) = 0x00000000  libraries/gadtools.h: *82
TEXTPEN            #define (0x0002) = 0x00000002  intuition/screens.h: *84
TEXT_KIND          #define 13 = 0x0000000d  libraries/gadtools.h: *47
TFCH_ID            #define 0x0f02 = 0x00000f02  libraries/diskfont.h: *52
TF_ETASK           #define (1<<3) = 0x00000008  exec/tasks.h: *60
TF_EXCEPT          #define (1<<5) = 0x00000020  exec/tasks.h: *62
TF_LAUNCH          #define (1<<7) = 0x00000080  exec/tasks.h: *64
TF_PROCTIME        #define (1<<0) = 0x00000001  exec/tasks.h: *59
TF_STACKCHK        #define (1<<4) = 0x00000010  exec/tasks.h: *61
TF_SWITCH          #define (1<<6) = 0x00000040  exec/tasks.h: *63
TFontContents      structure tag size 0x0104 libraries/diskfont.h: *37
TICKS_PER_SECOND   #define 50 = 0x00000032  dos/dos.h: *60
TIMERNAME          #define "timer.device"  devices/timer.h: *25
TOBB               macro (1 argument)   graphics/gfx.h: *24
TODA_SAFE          #define 8 = 0x00000008  graphics/gfxbase.h: *98
TOF_WaitQ          struct List(size 0x000e bytes) in struct GfxBase
           +0x00c0 graphics/gfxbase.h: *52
```

Include File Cross Reference Page 131

```
TOGGLESELECT      #define GACT_TOGGLESELECT = 0x00000100
                  intuition/iobsolete.h: *78
TOPAZ_EIGHTY      #define 8 = 0x00000008  intuition/preferences.h: *41
TOPAZ_SIXTY       #define 9 = 0x00000009  intuition/preferences.h: *42
TOPBORDER         #define GACT_TOPBORDER = 0x00000040
                  intuition/iobsolete.h: *75
TOPHIT            #define 1 = 0x00000001  graphics/collide.h: *32
TO_MONITOR        #define 0 = 0x00000000  graphics/monitor.h: *56
TRUE              #define 1 = 0x00000001  exec/types.h: *70
TR_ADDREQUEST     #define CMD_NONSTD = 0x00000009  devices/timer.h: *43
TR_GETSYSTIME     #define (CMD_NONSTD+1) = 0x0000000a  devices/timer.h: *44
TR_MakeBad        #define -4 = 0xfffffffc  libraries/translator.h: *20
TR_NoMem          #define -2 = 0xfffffffe  libraries/translator.h: *19
TR_NotUsed        #define -1 = 0xffffffff  libraries/translator.h: *18
TR_SETSYSTIME     #define (CMD_NONSTD+2) = 0x0000000b  devices/timer.h: *45
TS_ADDED          #define 1 = 0x00000001  exec/tasks.h: *68
TS_EXCEPT         #define 5 = 0x00000005  exec/tasks.h: *72
TS_INVALID        #define 0 = 0x00000000  exec/tasks.h: *67
TS_READY          #define 3 = 0x00000003  exec/tasks.h: *70
TS_REMOVED        #define 6 = 0x00000006  exec/tasks.h: *73
TS_RUN            #define 2 = 0x00000002  exec/tasks.h: *69
TS_WAIT           #define 4 = 0x00000004  exec/tasks.h: *71
TTextAttr         structure tag size 0x000c graphics/text.h: *74
                  libraries/diskfont.h: 102
TWO_PI            #define (((double) 2) * PI)  libraries/mathffp.h: *18
                  libraries/mathieeedp.h: *18
Tag               typedef ULONG  utility/tagitem.h: *30, 33
TagItem           structure tag size 0x0008 graphics/view.h: *128
                  utility/tagitem.h: 32
                  graphics/text.h: 79, 127
                  intuition/intuition.h: 1082
                  intuition/screens.h: 357
                  intuition/classusr.h: 80, 90
Task              structure tag size 0x005c exec/tasks.h: *26
                  exec/semaphores.h: 37, 46
                  exec/execbase.h: 63
                  dos/dosextens.h: 40
                  intuition/screens.h: 388
                  devices/prtbase.h: 93
                  dos/notify.h: 66
                  graphics/gfxbase.h: 51
                  resources/disk.h: 62
TaskExceptCode    pointer to void in struct ExecBase
          +0x0134 exec/execbase.h: *78
TaskExitCode      pointer to void in struct ExecBase
          +0x0138 exec/execbase.h: *79
TaskReady         struct List(size 0x000e bytes) in struct ExecBase
          +0x0196 exec/execbase.h: *92
TaskSigAlloc      unsigned long int in struct ExecBase
          +0x013c exec/execbase.h: *80
TaskTrapAlloc     unsigned short int in struct ExecBase
          +0x0140 exec/execbase.h: *81
TaskTrapCode      pointer to void in struct ExecBase
          +0x0130 exec/execbase.h: *77
TaskWait          struct List(size 0x000e bytes) in struct ExecBase
          +0x01a4 exec/execbase.h: *93
TermArray0        unsigned long int in struct IOTArray
          +0x0000 devices/serial.h: *23
TermArray1        unsigned long int in struct IOTArray
          +0x0004 devices/serial.h: *24
TextAttr          structure tag size 0x0008 graphics/text.h: *67
                  intuition/intuition.h: 576
                  intuition/screens.h: 124, 320, 352
                  libraries/asl.h: 148
                  libraries/diskfont.h: 97
                  libraries/gadtools.h: 99
```

Include File Cross Reference Page 132

```
TextExtent        structure tag size 0x000c graphics/text.h: *166
TextFont          structure tag size 0x0034 devices/conunit.h: *89
                  graphics/rastport.h: 78
                  graphics/text.h: 91, 125, 153
                  intuition/intuition.h: 895
                  intuition/screens.h: 67
                  graphics/gfxbase.h: 38
                  intuition/sghooks.h: 21
                  libraries/diskfont.h: 77
TextFontExtension structure tag size 0x0018 graphics/text.h: *121
TextFonts         struct List(size 0x000e bytes) in struct GfxBase
          +0x008c graphics/gfxbase.h: *37
ThisTask          pointer to struct Task in struct ExecBase
          +0x0114 exec/execbase.h: *63
TimeSet           short int in struct AnimComp +0x0004 graphics/gels.h: *180
Timer             short int in struct AnimComp +0x0002 graphics/gels.h: *176
Title             pointer to unsigned char in struct Window
          +0x0020 intuition/intuition.h: *811
Title             pointer to unsigned char in struct NewWindow
          +0x001a intuition/intuition.h: *998
Title             pointer to unsigned char in struct ExtNewWindow
          +0x001a intuition/intuition.h: *1056
Title             pointer to unsigned char in struct Screen
          +0x0016 intuition/screens.h: *110
TmpRas            structure tag size 0x0008 graphics/rastport.h: *34, 61
TmpRas            pointer to struct TmpRas in struct RastPort
          +0x000c graphics/rastport.h: *61
Top               short int in struct IBox +0x0002 intuition/intuition.h: *785
TopBorder         unsigned short int in struct PropInfo
          +0x0014 intuition/intuition.h: *492
TopEdge           short int in struct Menu +0x0006 intuition/intuition.h: *65
TopEdge           short int in struct MenuItem
          +0x0006 intuition/intuition.h: *93
TopEdge           short int in struct Requester
          +0x0006 intuition/intuition.h: *149
TopEdge           short int in struct Gadget +0x0006 intuition/intuition.h: *220
TopEdge           short int in struct IntuiText
          +0x0006 intuition/intuition.h: *575
TopEdge           short int in struct Border +0x0002 intuition/intuition.h: *600
TopEdge           short int in struct Image +0x0002 intuition/intuition.h: *622
TopEdge           short int in struct Window +0x0006 intuition/intuition.h: *799
TopEdge           short int in struct NewWindow
          +0x0002 intuition/intuition.h: *976
TopEdge           short int in struct ExtNewWindow
          +0x0002 intuition/intuition.h: *1046
TopEdge           short int in struct Screen +0x000a intuition/screens.h: *103
TopEdge           short int in struct NewScreen
          +0x0002 intuition/screens.h: *312
TopEdge           short int in struct ExtNewScreen
          +0x0002 intuition/screens.h: *348
TotalColorClocks unsigned short int in struct MonitorInfo
          +0x0026 graphics/displayinfo.h: *117
TotalRows         unsigned short int in struct MonitorInfo
          +0x0024 graphics/displayinfo.h: *116
TransparencyBits pointer to unsigned short int in struct ColorMap
          +0x000c graphics/view.h: *121
TransparencyPlane unsigned char in struct ColorMap
          +0x0010 graphics/view.h: *122
TxBaseline        unsigned short int in struct RastPort
          +0x003e graphics/rastport.h: *83
TxFlags           unsigned char in struct RastPort
          +0x0039 graphics/rastport.h: *80
TxHeight          unsigned short int in struct RastPort
          +0x003a graphics/rastport.h: *81
TxSpacing         short int in struct RastPort +0x0040 graphics/rastport.h: *84
TxWidth           unsigned short int in struct RastPort
```

Include File Cross Reference Page 133

```
        +0x003c graphics/rastport.h: *82
TxtOScan        struct Rectangle(size 0x0008 bytes) in struct DimensionInfo
        +0x0032 graphics/displayinfo.h: *103
Type            unsigned char in struct ColorMap +0x0001 graphics/view.h: *117
Type            unsigned short int in struct NewWindow
        +0x002e intuition/intuition.h: *1033
Type            unsigned short int in struct ExtNewWindow
        +0x002e intuition/intuition.h: *1071
Type            unsigned short int in struct NewScreen
        +0x000e intuition/screens.h: *318
Type            unsigned short int in struct ExtNewScreen
        +0x000e intuition/screens.h: *351
u1              union (no tag) (size 0x0002 bytes) in struct (no tag)
        +0x0000 graphics/copper.h: *38
u2              union (no tag) (size 0x0002 bytes) in struct (no tag)
        +0x0002 graphics/copper.h: *43
u3              union (no tag) (size 0x0004 bytes) in struct CopIns
        +0x0002 graphics/copper.h: *45
u4              struct (no tag) (size 0x0004 bytes) in union (no tag)
        +0x0000 graphics/copper.h: *44
unit_MsgPort    struct MsgPort(size 0x0022 bytes) in struct Unit
        +0x0000 exec/devices.h: *34
unit_OpenCnt    unsigned short int in struct Unit +0x0024 exec/devices.h: *38
unit_flags      unsigned char in struct Unit +0x0022 exec/devices.h: *36
unit_pad        unsigned char in struct Unit +0x0023 exec/devices.h: *37
unusedreg       unsigned char in struct CIA +0x0b00 hardware/cia.h: *54
UBYTE           unsigned char  exec/types.h: *49
UCOUNT          unsigned short int  exec/types.h: *59
UCopIns         pointer to struct UCopList in struct ViewPort
        +0x0014 graphics/view.h: *49
UCopList        structure tag size 0x000c graphics/copper.h: *83, 85
                graphics/view.h: 49
ULONG           unsigned long int  exec/types.h: *39
UNITF_ACTIVE    #define (1<<0) = 0x00000001  exec/devices.h: *42
UNITF_INTASK    #define (1<<1) = 0x00000002  exec/devices.h: *43
UNIT_ECLOCK     #define 2 = 0x00000002  devices/timer.h: *21
UNIT_MICROHZ    #define 0 = 0x00000000  devices/timer.h: *19
UNIT_VBLANK     #define 1 = 0x00000001  devices/timer.h: *20
UNIT_WAITECLOCK #define 4 = 0x00000004  devices/timer.h: *23
UNIT_WAITUNTIL  #define 3 = 0x00000003  devices/timer.h: *22
UPFRONTGADGET   #define 0 = 0x00000000  intuition/intuitionbase.h: *47
UPIMAGE         #define (0x0BL) = 0x0000000b  intuition/imageclass.h: *107
USEREQIMAGE     #define 0x0020 = 0x00000020  intuition/intuition.h: *194
USER_COPPER_CLIP #define 0x20 = 0x00000020  graphics/view.h: *145
USE_BPLCON3     #define 1 = 0x00000001  hardware/custom.h: *162
USHORT          unsigned short int  exec/types.h: *57
US_LEGAL        #define 0x10 = 0x00000010  intuition/preferences.h: *186
US_LETTER       #define 0x00 = 0x00000000  intuition/preferences.h: *185
UTILITY_DATE_H  #define 1 = 0x00000001  utility/date.h: *2
UTILITY_HOOKS_H #define TRUE = 0x00000001  utility/hooks.h: *2, 1
                dos/exall.h: 19
                intuition/classes.h: 15
                intuition/classusr.h: 16
                libraries/asl.h: 28
UTILITY_TAGITEM_H #define TRUE = 0x00000001  utility/tagitem.h: *2, 1
                intuition/intuition.h: 51
                intuition/screens.h: 39
                dos/dostags.h: 17
                graphics/displayinfo.h: 27
                graphics/videocontrol.h: 19
                intuition/gadgetclass.h: 23
                intuition/icclass.h: 16
                intuition/imageclass.h: 16
                libraries/asl.h: 32
                libraries/gadtools.h: 21
UWORD           unsigned short int  exec/types.h: *42
```

Include File Cross Reference Page 134

```
UnReadChar      macro (1 argument)   dos/stdio.h: *19
UndoBuffer      pointer to unsigned char in struct StringInfo
        +0x0004 intuition/intuition.h: *528
UndoPos         short int in struct StringInfo
        +0x000e intuition/intuition.h: *534
Unit            structure tag size 0x0026 exec/devices.h: *33
                exec/io.h: 23, 32
                devices/printer.h: 145, 159
                devices/trackdisk.h: 241
UserData        pointer to void in struct Gadget
        +0x0028 intuition/intuition.h: *261
UserData        pointer to char in struct Window
        +0x0078 intuition/intuition.h: *885
UserData        pointer to unsigned char in struct Screen
        +0x0156 intuition/screens.h: *149
UserPort        pointer to struct MsgPort in struct Window
        +0x0056 intuition/intuition.h: *856
UtilityBase     pointer to unsigned long int in struct GfxBase
        +0x019e graphics/gfxbase.h: *91
vblank          struct AnalogSignalInterval(size 0x0004 bytes) in struct
                SpecialMonitor
        +0x002e graphics/monitor.h: *151
vbsrv           struct Interrupt(size 0x0016 bytes) in struct GfxBase
        +0x004a graphics/gfxbase.h: *36
vbstop          unsigned short int in struct Custom
        +0x01ce hardware/custom.h: *130
vbstrt          unsigned short int in struct Custom
        +0x01cc hardware/custom.h: *129
vhposr          unsigned short int in struct Custom
        +0x0006 hardware/custom.h: *31
vhposw          unsigned short int in struct Custom
        +0x002c hardware/custom.h: *49
voice           struct narrator_rb(size 0x0058 bytes) in struct mouth_rb
        +0x0000 devices/narrator.h: *129
volume          unsigned short int in struct narrator_rb
        +0x003e devices/narrator.h: *101
vposr           unsigned short int in struct Custom
        +0x0004 hardware/custom.h: *30
vposw           unsigned short int in struct Custom
        +0x002a hardware/custom.h: *48
vsstop          unsigned short int in struct Custom
        +0x01ca hardware/custom.h: *128
vsstrt          unsigned short int in struct Custom
        +0x01e0 hardware/custom.h: *139
vsync           struct AnalogSignalInterval(size 0x0004 bytes) in struct
                SpecialMonitor
        +0x0036 graphics/monitor.h: *153
vsync_hblank    array [2] of unsigned short int in struct copinit
        +0x0000 graphics/copper.h: *94
vtotal          unsigned short int in struct Custom
        +0x01c8 hardware/custom.h: *127
VANILLAKEY      #define IDCMP_VANILLAKEY = 0x00200000
                intuition/iobsolete.h: *135
VARBEAM         #define 0x0080 = 0x00000080  hardware/custom.h: *152
VARCSYNC        #define 0x0010 = 0x00000010  hardware/custom.h: *155
VARHSYNC        #define 0x0100 = 0x00000100  hardware/custom.h: *151
VARVBLANK       #define 0x1000 = 0x00001000  hardware/custom.h: *147
VARVSYNC        #define 0x0200 = 0x00000200  hardware/custom.h: *150
VBlank          char in struct GfxBase +0x00a0 graphics/gfxbase.h: *40
VBlankFrequency unsigned char in struct ExecBase +0x0212 exec/execbase.h: *111
VGA70_BEAMCON   #define (SPECIAL_BEAMCON ^ VSYNCTRUE) = 0x00001a8a
                graphics/monitor.h: *121
VGA70_COLORCLOCKS #define (STANDARD_COLORCLOCKS/2) = 0x00000071
                graphics/monitor.h: *108
VGA70_DENISE_MIN #define 59 = 0x0000003b  graphics/monitor.h: *110
VGA70_HBSTOP    #define 0x1E = 0x0000001e  graphics/monitor.h: *115
```

```
VGA70_HBSTRT      #define 0x08 = 0x00000008  graphics/monitor.h: *112
VGA70_HSSTOP      #define 0x1C = 0x0000001c  graphics/monitor.h: *114
VGA70_HSSTRT      #define 0x0E = 0x0000000e  graphics/monitor.h: *113
VGA70_MONITOR_NAME #define "vga70.monitor"  graphics/monitor.h: *122
VGA70_TOTAL_ROWS #define 449 = 0x000001c1  graphics/monitor.h: *109
VGA70_VBSTOP      #define 0x0F73 = 0x00000f73  graphics/monitor.h: *119
VGA70_VBSTRT      #define 0x0000 = 0x00000000  graphics/monitor.h: *116
VGA70_VSSTOP      #define 0x0388 = 0x00000388  graphics/monitor.h: *118
VGA70_VSSTRT      #define 0x02A6 = 0x000002a6  graphics/monitor.h: *117
VGAEXTRAHALFBRITELACE_KEY #define 0x00031085 = 0x00031085
                  graphics/displayinfo.h: *203
VGAEXTRAHALFBRITE_KEY #define 0x00031084 = 0x00031084
                  graphics/displayinfo.h: *202
VGAEXTRALORESDPF2_KEY #define 0x00031444 = 0x00031444
                  graphics/displayinfo.h: *196
VGAEXTRALORESDPF_KEY #define 0x00031404 = 0x00031404
                  graphics/displayinfo.h: *190
VGAEXTRALORESLACEDPF2_KEY #define 0x00031445 = 0x00031445
                  graphics/displayinfo.h: *199
VGAEXTRALORESLACEDPF_KEY #define 0x00031405 = 0x00031405
                  graphics/displayinfo.h: *193
VGAEXTRALORESLACE_KEY #define 0x00031005 = 0x00031005
                  graphics/displayinfo.h: *186
VGAEXTRALORES_KEY #define 0x00031004 = 0x00031004
                  graphics/displayinfo.h: *182
VGAHAMLACE_KEY    #define 0x00031805 = 0x00031805  graphics/displayinfo.h: *189
VGAHAM_KEY        #define 0x00031804 = 0x00031804  graphics/displayinfo.h: *185
VGALORESDPF2_KEY #define 0x00039444 = 0x00039444  graphics/displayinfo.h: *197
VGALORESDPF_KEY   #define 0x00039404 = 0x00039404  graphics/displayinfo.h: *191
VGALORESLACEDPF2_KEY #define 0x00039445 = 0x00039445
                  graphics/displayinfo.h: *200
VGALORESLACEDPF_KEY #define 0x00039405 = 0x00039405
                  graphics/displayinfo.h: *194
VGALORESLACE_KEY #define 0x00039005 = 0x00039005  graphics/displayinfo.h: *187
VGALORES_KEY      #define 0x00039004 = 0x00039004  graphics/displayinfo.h: *183
VGAPRODUCTDPF2_KEY #define 0x00039464 = 0x00039464
                  graphics/displayinfo.h: *198
VGAPRODUCTDPF_KEY #define 0x00039424 = 0x00039424
                  graphics/displayinfo.h: *192
VGAPRODUCTLACEDPF2_KEY #define 0x00039465 = 0x00039465
                  graphics/displayinfo.h: *201
VGAPRODUCTLACEDPF_KEY #define 0x00039425 = 0x00039425
                  graphics/displayinfo.h: *195
VGAPRODUCTLACE_KEY #define 0x00039025 = 0x00039025
                  graphics/displayinfo.h: *188
VGAPRODUCT_KEY    #define 0x00039024 = 0x00039024  graphics/displayinfo.h: *184
VGA_COLORCLOCKS   #define (STANDARD_COLORCLOCKS/2) = 0x00000071
                  graphics/monitor.h: *93
VGA_DENISE_MIN    #define 59 = 0x0000003b  graphics/monitor.h: *95
VGA_HBSTOP        #define 0x1E = 0x0000001e  graphics/monitor.h: *100
VGA_HBSTRT        #define 0x08 = 0x00000008  graphics/monitor.h: *97
VGA_HSSTOP        #define 0x1C = 0x0000001c  graphics/monitor.h: *99
VGA_HSSTRT        #define 0x0E = 0x0000000e  graphics/monitor.h: *98
VGA_MONITOR_ID    #define 0x00031000 = 0x00031000  graphics/displayinfo.h: *180
VGA_MONITOR_NAME #define "vga.monitor"  graphics/monitor.h: *106
VGA_TOTAL_ROWS    #define (STANDARD_NTSC_ROWS*2) = 0x0000020c
                  graphics/monitor.h: *94
VGA_VBSTOP        #define 0x0CCD = 0x00000ccd  graphics/monitor.h: *104
VGA_VBSTRT        #define 0x0000 = 0x00000000  graphics/monitor.h: *101
VGA_VSSTOP        #define 0x0235 = 0x00000235  graphics/monitor.h: *103
VGA_VSSTRT        #define 0x0153 = 0x00000153  graphics/monitor.h: *102
VIDEOCONTROL_BATCH #define 0x10 = 0x00000010  graphics/view.h: *144
VIEWPORT_EXTRA_TYPE #define 2 = 0x00000002  graphics/gfxnodes.h: *34
VIEW_EXTRA_TYPE   #define 1 = 0x00000001  graphics/gfxnodes.h: *33
VISITOR           #define WFLG_VISITOR = 0x08000000
                  intuition/iobsolete.h: *170
```

```
VOID              #define void  exec/types.h: *27
VPF_A2024         #define 0x40 = 0x00000040  graphics/view.h: *103
VPF_AGNUS         #define 0x20 = 0x00000020  graphics/view.h: *104
VPF_TENHZ         #define 0x20 = 0x00000020  graphics/view.h: *105
VPModeID          unsigned long int in struct ColorMap
          +0x0024 graphics/view.h: *129
VPOSRLOF          #define 0x8000 = 0x00008000  graphics/display.h: *40
VP_HIDE           #define 0x2000 = 0x00002000  graphics/view.h: *99
VPotRes           unsigned short int in struct PropInfo
          +0x0010 intuition/intuition.h: *490
VSBob             pointer to struct Bob in struct VSprite
          +0x0034 graphics/gels.h: *116
VSIZEBITS         #define 16-HSIZEBITS = 0x0000000a  hardware/blit.h: *16
VSIZEMASK         #define 0x3FF = 0x000003ff  hardware/blit.h: *18
VSOVERFLOW        #define 0x0800 = 0x00000800  graphics/gels.h: *30
VSPRITE           #define 0x0001 = 0x00000001  graphics/gels.h: *22
VSYNCTRUE         #define 0x0002 = 0x00000002  hardware/custom.h: *158
VSprite           structure tag size 0x003c graphics/rastport.h: *46
                  graphics/gels.h: 70, 74, 75, 81, 82, 156, 233
VTAG_ATTACH_CM_GET #define 0x8000001B = 0x8000001b
                  graphics/videocontrol.h: *51
VTAG_ATTACH_CM_SET #define 0x8000000B = 0x8000000b
                  graphics/videocontrol.h: *35
VTAG_BATCH_CM_CLR #define 0x8000000D = 0x8000000d
                  graphics/videocontrol.h: *37
VTAG_BATCH_CM_GET #define 0x8000001C = 0x8000001c
                  graphics/videocontrol.h: *52
VTAG_BATCH_CM_SET #define 0x8000000E = 0x8000000e
                  graphics/videocontrol.h: *38
VTAG_BATCH_ITEMS_ADD #define 0x8000001F = 0x8000001f
                  graphics/videocontrol.h: *55
VTAG_BATCH_ITEMS_GET #define 0x8000001D = 0x8000001d
                  graphics/videocontrol.h: *53
VTAG_BATCH_ITEMS_SET #define 0x8000001E = 0x8000001e
                  graphics/videocontrol.h: *54
VTAG_BITPLANEKEY_CLR #define 0x80000002 = 0x80000002
                  graphics/videocontrol.h: *26
VTAG_BITPLANEKEY_GET #define 0x80000016 = 0x80000016
                  graphics/videocontrol.h: *46
VTAG_BITPLANEKEY_SET #define 0x80000003 = 0x80000003
                  graphics/videocontrol.h: *27
VTAG_BORDERBLANK_CLR #define 0x80000004 = 0x80000004
                  graphics/videocontrol.h: *28
VTAG_BORDERBLANK_GET #define 0x80000017 = 0x80000017
                  graphics/videocontrol.h: *47
VTAG_BORDERBLANK_SET #define 0x80000005 = 0x80000005
                  graphics/videocontrol.h: *29
VTAG_BORDERNOTRANS_CLR #define 0x80000006 = 0x80000006
                  graphics/videocontrol.h: *30
VTAG_BORDERNOTRANS_GET #define 0x80000018 = 0x80000018
                  graphics/videocontrol.h: *48
VTAG_BORDERNOTRANS_SET #define 0x80000007 = 0x80000007
                  graphics/videocontrol.h: *31
VTAG_CHROMAKEY_CLR #define 0x80000000 = 0x80000000
                  graphics/videocontrol.h: *24
VTAG_CHROMAKEY_GET #define 0x80000015 = 0x80000015
                  graphics/videocontrol.h: *45
VTAG_CHROMAKEY_SET #define 0x80000001 = 0x80000001
                  graphics/videocontrol.h: *25
VTAG_CHROMA_PEN_CLR #define 0x80000008 = 0x80000008
                  graphics/videocontrol.h: *32
VTAG_CHROMA_PEN_GET #define 0x80000019 = 0x80000019
                  graphics/videocontrol.h: *49
VTAG_CHROMA_PEN_SET #define 0x80000009 = 0x80000009
                  graphics/videocontrol.h: *33
VTAG_CHROMA_PLANE_GET #define 0x8000001A = 0x8000001a
```

Include File Cross Reference Page 137

```
                graphics/videocontrol.h: *50
VTAG_CHROMA_PLANE_SET #define 0x8000000A = 0x8000000a
                graphics/videocontrol.h: *34
VTAG_COERCE_DISP_GET #define 0x80000011 = 0x80000011
                graphics/videocontrol.h: *41
VTAG_COERCE_DISP_SET #define 0x80000012 = 0x80000012
                graphics/videocontrol.h: *42
VTAG_END_CM     #define 0x00000000 = 0x00000000  graphics/videocontrol.h: *23
VTAG_NEXTBUF_CM #define 0x8000000C = 0x8000000c  graphics/videocontrol.h: *36
VTAG_NORMAL_DISP_GET #define 0x8000000F = 0x8000000f
                graphics/videocontrol.h: *39
VTAG_NORMAL_DISP_SET #define 0x80000010 = 0x80000010
                graphics/videocontrol.h: *40
VTAG_USERCLIP_CLR #define 0x80000025 = 0x80000025
                graphics/videocontrol.h: *61
VTAG_USERCLIP_GET #define 0x80000023 = 0x80000023
                graphics/videocontrol.h: *59
VTAG_USERCLIP_SET #define 0x80000024 = 0x80000024
                graphics/videocontrol.h: *60
VTAG_VIEWPORTEXTRA_GET #define 0x80000013 = 0x80000013
                graphics/videocontrol.h: *43
VTAG_VIEWPORTEXTRA_SET #define 0x80000014 = 0x80000014
                graphics/videocontrol.h: *44
VTAG_VPMODEID_CLR #define 0x80000022 = 0x80000022
                graphics/videocontrol.h: *58
VTAG_VPMODEID_GET #define 0x80000020 = 0x80000020
                graphics/videocontrol.h: *56
VTAG_VPMODEID_SET #define 0x80000021 = 0x80000021
                graphics/videocontrol.h: *57
VUserExt        short int in struct VSprite +0x003a graphics/gels.h: *133
VUserStuff      #define WORD  graphics/gels.h: *56, 133
VWAITPOS        #define u3.u4.u1.VWaitPos  graphics/copper.h: *49
VWaitPos        short int in union (no tag)  +0x0000 graphics/copper.h: *36
VWritef         macro (2 arguments)   dos/stdio.h: *24
VctrPtr         pointer to short int in struct AreaInfo
        +0x0004 graphics/rastport.h: *26
VctrTbl         pointer to short int in struct AreaInfo
        +0x0000 graphics/rastport.h: *25
VertBody        unsigned short int in struct PropInfo
        +0x0008 intuition/intuition.h: *485
VertPot         unsigned short int in struct PropInfo
        +0x0004 intuition/intuition.h: *465
VideoOScan      struct Rectangle(size 0x0008 bytes) in struct DimensionInfo
        +0x002a graphics/displayinfo.h: *102
View            structure tag size 0x0012 graphics/view.h: *58, 73
                graphics/gfxbase.h: 28
                intuition/intuitionbase.h: 72
View            pointer to struct View in struct ViewExtra
        +0x0018 graphics/view.h: *73
ViewExtra       structure tag size 0x0020 graphics/view.h: *70
ViewInitX       short int in struct Preferences
        +0x0078 intuition/preferences.h: *78
ViewInitY       short int in struct Preferences
        +0x007a intuition/preferences.h: *78
ViewLord        struct View(size 0x0012 bytes) in struct IntuitionBase
        +0x0022 intuition/intuitionbase.h: *72
ViewModes       unsigned short int in struct NewScreen
        +0x000c intuition/screens.h: *316
ViewModes       unsigned short int in struct ExtNewScreen
        +0x000c intuition/screens.h: *350
ViewPort        structure tag size 0x0028 graphics/copper.h: *67
                graphics/view.h: 41, 43, 60, 82, 125
                intuition/screens.h: 127
ViewPort        pointer to struct ViewPort in struct View
        +0x0000 graphics/view.h: *60
ViewPort        pointer to struct ViewPort in struct ViewPortExtra
```

Include File Cross Reference Page 138

```
        +0x0018 graphics/view.h: *82
ViewPort        struct ViewPort(size 0x0028 bytes) in struct Screen
        +0x002c intuition/screens.h: *127
ViewPortExtra   structure tag size 0x0024 graphics/view.h: *79, 120
ViewPosition    struct tPoint(size 0x0004 bytes) in struct MonitorInfo
        +0x0014 graphics/displayinfo.h: *113
ViewPositionRange struct Rectangle(size 0x0008 bytes) in struct MonitorInfo
        +0x001c graphics/displayinfo.h: *115
ViewResolution  struct tPoint(size 0x0004 bytes) in struct MonitorInfo
        +0x0018 graphics/displayinfo.h: *114
ViewXOffset     char in struct Preferences
        +0x0076 intuition/preferences.h: *76
ViewYOffset     char in struct Preferences
        +0x0077 intuition/preferences.h: *77
wa_Lock         long int in struct WBArg +0x0000 workbench/startup.h: *37
wa_Name         pointer to char in struct WBArg
        +0x0004 workbench/startup.h: *38
wait14          array [2] of unsigned short int in struct copinit
        +0x0054 graphics/copper.h: *98
wait_forever    array [2] of unsigned short int in struct copinit
        +0x006c graphics/copper.h: *102
wb_Depth        unsigned char in struct Preferences
        +0x00e6 intuition/preferences.h: *125
wb_Height       unsigned short int in struct Preferences
        +0x00e4 intuition/preferences.h: *124
wb_Width        unsigned short int in struct Preferences
        +0x00e2 intuition/preferences.h: *123
wday            unsigned short int in struct ClockData
        +0x000c utility/date.h: *26
width           unsigned char in struct mouth_rb
        +0x0058 devices/narrator.h: *130
wordreserved    array [7] of unsigned short int in struct RastPort
        +0x004e graphics/rastport.h: *88
WA_Activate     #define (WA_Dummy + 0x26) = 0x80000089
                intuition/intuition.h: *1189
WA_AutoAdjust   #define (WA_Dummy + 0x2D) = 0x80000090
                intuition/intuition.h: *1200
WA_BackFill     #define (WA_Dummy + 0x1C) = 0x8000007f
                intuition/intuition.h: *1171
WA_Backdrop     #define (WA_Dummy + 0x22) = 0x80000085
                intuition/intuition.h: *1185
WA_BlockPen     #define (WA_Dummy + 0x06) = 0x80000069
                intuition/intuition.h: *1100
WA_Borderless   #define (WA_Dummy + 0x25) = 0x80000088
                intuition/intuition.h: *1188
WA_Checkmark    #define (WA_Dummy + 0x0A) = 0x8000006d
                intuition/intuition.h: *1105
WA_CloseGadget  #define (WA_Dummy + 0x21) = 0x80000084
                intuition/intuition.h: *1184
WA_Colors       #define (WA_Dummy + 0x19) = 0x8000007c
                intuition/intuition.h: *1151
WA_CustomScreen #define (WA_Dummy + 0x0D) = 0x80000070
                intuition/intuition.h: *1111
WA_DepthGadget  #define (WA_Dummy + 0x20) = 0x80000083
                intuition/intuition.h: *1183
WA_DetailPen    #define (WA_Dummy + 0x05) = 0x80000068
                intuition/intuition.h: *1099
WA_DragBar      #define (WA_Dummy + 0x1F) = 0x80000082
                intuition/intuition.h: *1182
WA_Dummy        #define (TAG_USER + 99) = 0x80000063
                intuition/intuition.h: *1092
WA_Flags        #define (WA_Dummy + 0x08) = 0x8000006b
                intuition/intuition.h: *1103
WA_Gadgets      #define (WA_Dummy + 0x09) = 0x8000006c
                intuition/intuition.h: *1104
WA_GimmeZeroZero #define (WA_Dummy + 0x2E) = 0x80000091
```

Include File Cross Reference Page 139

```
                   intuition/intuition.h: *1205
WA_Height          #define (WA_Dummy + 0x04) = 0x80000067
                   intuition/intuition.h: *1098
WA_IDCMP           #define (WA_Dummy + 0x07) = 0x8000006a
                   intuition/intuition.h: *1101
WA_InnerHeight     #define (WA_Dummy + 0x14) = 0x80000077
                   intuition/intuition.h: *1122
WA_InnerWidth      #define (WA_Dummy + 0x13) = 0x80000076
                   intuition/intuition.h: *1121
WA_Left            #define (WA_Dummy + 0x01) = 0x80000064
                   intuition/intuition.h: *1095
WA_MaxHeight       #define (WA_Dummy + 0x12) = 0x80000075
                   intuition/intuition.h: *1117
WA_MaxWidth        #define (WA_Dummy + 0x11) = 0x80000074
                   intuition/intuition.h: *1116
WA_MenuHelp        #define (WA_Dummy + 0x2F) = 0x80000092
                   intuition/intuition.h: *1209
WA_MinHeight       #define (WA_Dummy + 0x10) = 0x80000073
                   intuition/intuition.h: *1115
WA_MinWidth        #define (WA_Dummy + 0x0F) = 0x80000072
                   intuition/intuition.h: *1114
WA_MouseQueue      #define (WA_Dummy + 0x1B) = 0x8000007e
                   intuition/intuition.h: *1167
WA_NoCareRefresh   #define (WA_Dummy + 0x24) = 0x80000087
                   intuition/intuition.h: *1187
WA_PubScreen       #define (WA_Dummy + 0x16) = 0x80000079
                   intuition/intuition.h: *1136
WA_PubScreenFallBack #define (WA_Dummy + 0x17) = 0x8000007a
                   intuition/intuition.h: *1143
WA_PubScreenName   #define (WA_Dummy + 0x15) = 0x80000078
                   intuition/intuition.h: *1131
WA_RMBTrap         #define (WA_Dummy + 0x27) = 0x8000008a
                   intuition/intuition.h: *1190
WA_ReportMouse     #define (WA_Dummy + 0x23) = 0x80000086
                   intuition/intuition.h: *1186
WA_RptQueue        #define (WA_Dummy + 0x1D) = 0x80000080
                   intuition/intuition.h: *1175
WA_ScreenTitle     #define (WA_Dummy + 0x0C) = 0x8000006f
                   intuition/intuition.h: *1110
WA_SimpleRefresh   #define (WA_Dummy + 0x29) = 0x8000008c
                   intuition/intuition.h: *1192
WA_SizeBBottom     #define (WA_Dummy + 0x2C) = 0x8000008f
                   intuition/intuition.h: *1197
WA_SizeBRight      #define (WA_Dummy + 0x2B) = 0x8000008e
                   intuition/intuition.h: *1196
WA_SizeGadget      #define (WA_Dummy + 0x1E) = 0x80000081
                   intuition/intuition.h: *1181
WA_SmartRefresh    #define (WA_Dummy + 0x2A) = 0x8000008d
                   intuition/intuition.h: *1194
WA_SuperBitMap     #define (WA_Dummy + 0x0E) = 0x80000071
                   intuition/intuition.h: *1112
WA_Title           #define (WA_Dummy + 0x0B) = 0x8000006e
                   intuition/intuition.h: *1106
WA_Top             #define (WA_Dummy + 0x02) = 0x80000065
                   intuition/intuition.h: *1096
WA_WBenchWindow    #define (WA_Dummy + 0x28) = 0x8000008b
                   intuition/intuition.h: *1191
WA_Width           #define (WA_Dummy + 0x03) = 0x80000066
                   intuition/intuition.h: *1097
WA_WindowName      #define (WA_Dummy + 0x18) = 0x8000007b
                   intuition/intuition.h: *1149
WA_Zoom            #define (WA_Dummy + 0x1A) = 0x8000007d
                   intuition/intuition.h: *1159
WBAPPICON          #define 8 = 0x00000008  workbench/workbench.h: *42
WBArg              structure tag size 0x0008 workbench/startup.h: *33, 36
                   libraries/asl.h: 77
```

Include File Cross Reference Page 140

```
                   workbench/workbench.h: 132
WBDEVICE           #define 6 = 0x00000006  workbench/workbench.h: *40
WBDISK             #define 1 = 0x00000001  workbench/workbench.h: *35
WBDRAWER           #define 2 = 0x00000002  workbench/workbench.h: *36
WBENCHCLOSE        #define 0x0002 = 0x00000002  intuition/intuition.h: *779
WBENCHMESSAGE      #define IDCMP_WBENCHMESSAGE = 0x00020000
                   intuition/iobsolete.h: *131
WBENCHOPEN         #define 0x0001 = 0x00000001  intuition/intuition.h: *778
WBENCHSCREEN       #define 0x0001 = 0x00000001  intuition/screens.h: *161
WBENCHWINDOW       #define WFLG_WBENCHWINDOW = 0x02000000
                   intuition/iobsolete.h: *167
WBGARBAGE          #define 5 = 0x00000005  workbench/workbench.h: *39
WBKICK             #define 7 = 0x00000007  workbench/workbench.h: *41
WBPROJECT          #define 4 = 0x00000004  workbench/workbench.h: *38
WBStartup          structure tag size 0x0028 workbench/startup.h: *27
WBTOOL             #define 3 = 0x00000003  workbench/workbench.h: *37
WB_DISKMAGIC       #define 0xe310 = 0x0000e310  workbench/workbench.h: *77
WB_DISKREVISION    #define 1 = 0x00000001  workbench/workbench.h: *79
WB_DISKREVISIONMASK #define 255 = 0x000000ff  workbench/workbench.h: *81
WB_DISKVERSION     #define 1 = 0x00000001  workbench/workbench.h: *78
WBorBottom         char in struct Screen +0x0026 intuition/screens.h: *122
WBorLeft           char in struct Screen +0x0024 intuition/screens.h: *122
WBorRight          char in struct Screen +0x0025 intuition/screens.h: *122
WBorTop            char in struct Screen +0x0023 intuition/screens.h: *122
WDOWNBACK          #define GTYP_WDOWNBACK = 0x00000060
                   intuition/iobsolete.h: *101
WDRAGGING          #define GTYP_WDRAGGING = 0x00000020
                   intuition/iobsolete.h: *97
WFLG_ACTIVATE      #define 0x00001000 = 0x00001000  intuition/intuition.h: *932
WFLG_BACKDROP      #define 0x00000100 = 0x00000100  intuition/intuition.h: *924
WFLG_BORDERLESS    #define 0x00000800 = 0x00000800  intuition/intuition.h: *930
WFLG_CLOSEGADGET   #define 0x00000008 = 0x00000008  intuition/intuition.h: *911
WFLG_DEPTHGADGET   #define 0x00000004 = 0x00000004  intuition/intuition.h: *910
WFLG_DRAGBAR       #define 0x00000002 = 0x00000002  intuition/intuition.h: *909
WFLG_GIMMEZEROZERO #define 0x00000400 = 0x00000400
                   intuition/intuition.h: *928
WFLG_HASZOOM       #define 0x20000000 = 0x20000000  intuition/intuition.h: *957
WFLG_INREQUEST     #define 0x00004000 = 0x00004000  intuition/intuition.h: *937
WFLG_MENUSTATE     #define 0x00008000 = 0x00008000  intuition/intuition.h: *938
WFLG_NOCAREREFRESH #define 0x00020000 = 0x00020000
                   intuition/intuition.h: *942
WFLG_NW_EXTENDED   #define 0x00040000 = 0x00040000  intuition/intuition.h: *951
WFLG_OTHER_REFRESH #define 0x000000C0 = 0x000000c0
                   intuition/intuition.h: *922
WFLG_REFRESHBITS   #define 0x000000C0 = 0x000000c0  intuition/intuition.h: *918
WFLG_REPORTMOUSE   #define 0x00000200 = 0x00000200  intuition/intuition.h: *926
WFLG_RMBTRAP       #define 0x00010000 = 0x00010000  intuition/intuition.h: *941
WFLG_SIMPLE_REFRESH #define 0x00000040 = 0x00000040
                   intuition/intuition.h: *920
WFLG_SIZEBBOTTOM   #define 0x00000020 = 0x00000020  intuition/intuition.h: *914
WFLG_SIZEBRIGHT    #define 0x00000010 = 0x00000010  intuition/intuition.h: *913
WFLG_SIZEGADGET    #define 0x00000001 = 0x00000001  intuition/intuition.h: *908
WFLG_SMART_REFRESH #define 0x00000000 = 0x00000000
                   intuition/intuition.h: *919
WFLG_SUPER_BITMAP  #define 0x00000080 = 0x00000080  intuition/intuition.h: *921
WFLG_VISITOR       #define 0x08000000 = 0x08000000  intuition/intuition.h: *955
WFLG_WBENCHWINDOW  #define 0x02000000 = 0x02000000  intuition/intuition.h: *946
WFLG_WINDOWACTIVE  #define 0x00002000 = 0x00002000  intuition/intuition.h: *936
WFLG_WINDOWREFRESH #define 0x01000000 = 0x01000000
                   intuition/intuition.h: *945
WFLG_WINDOWTICKED  #define 0x04000000 = 0x04000000  intuition/intuition.h: *947
WFLG_ZOOMED        #define 0x10000000 = 0x10000000  intuition/intuition.h: *956
WINDOWACTIVE       #define WFLG_WINDOWACTIVE = 0x00002000
                   intuition/iobsolete.h: *161
WINDOWCLOSE        #define WFLG_CLOSEGADGET = 0x00000008
                   intuition/iobsolete.h: *148
```

Include File Cross Reference Page 141

```
WINDOWDEPTH      #define WFLG_DEPTHGADGET = 0x00000004
                 intuition/iobsolete.h: *147
WINDOWDRAG       #define WFLG_DRAGBAR = 0x00000002
                 intuition/iobsolete.h: *146
WINDOWREFRESH    #define WFLG_WINDOWREFRESH = 0x01000000
                 intuition/iobsolete.h: *166
WINDOWSIZING     #define WFLG_SIZEGADGET = 0x00000001
                 intuition/iobsolete.h: *145
WINDOWTICKED     #define WFLG_WINDOWTICKED = 0x04000000
                 intuition/iobsolete.h: *168
WLayer           pointer to struct Layer in struct Window
         +0x007c intuition/intuition.h: *890
WORD             short int  exec/types.h: *41
WORDBITS         unsigned short int  exec/types.h: *43
WORKBENCH_ICON_H #define  workbench/icon.h: *2
WORKBENCH_NAME   #define "workbench.library"  workbench/workbench.h: *119
WORKBENCH_STARTUP_H #define  workbench/startup.h: *2, 1
WORKBENCH_WORKBENCH_H #define  workbench/workbench.h: *2
WScreen          pointer to struct Screen in struct Window
         +0x002e intuition/intuition.h: *819
WUPFRONT         #define GTYP_WUPFRONT = 0x00000040
                 intuition/iobsolete.h: *99
W_TRACTOR        #define 0x30 = 0x00000030  intuition/preferences.h: *188
WarmCapture      pointer to void in struct ExecBase
         +0x0032 exec/execbase.h: *46
Width            short int in struct Layer +0x0086 graphics/clip.h: *57
Width            short int in struct Menu +0x0008 intuition/intuition.h: *66
Width            short int in struct MenuItem
         +0x0008 intuition/intuition.h: *94
Width            short int in struct Requester
         +0x0008 intuition/intuition.h: *150
Width            short int in struct Gadget +0x0008 intuition/intuition.h: *221
Width            short int in struct Image +0x0004 intuition/intuition.h: *623
Width            short int in struct IBox +0x0004 intuition/intuition.h: *786
Width            short int in struct Window +0x0008 intuition/intuition.h: *800
Width            short int in struct NewWindow
         +0x0004 intuition/intuition.h: *977
Width            short int in struct ExtNewWindow
         +0x0004 intuition/intuition.h: *1047
Width            short int in struct Screen +0x000c intuition/screens.h: *104
Width            short int in struct NewScreen
         +0x0004 intuition/screens.h: *312
Width            short int in struct ExtNewScreen
         +0x0004 intuition/screens.h: *348
Width            short int in struct VSprite +0x001c graphics/gels.h: *99
Width            short int in struct (no tag)
         +0x0000 intuition/imageclass.h: *165
Width            short int in struct (no tag)
         +0x0000 intuition/imageclass.h: *182
Width            short int in struct (no tag)
         +0x0000 intuition/imageclass.h: *197
Window           structure tag size 0x0088 devices/conunit.h: *58
                 intuition/intuition.h: 172, 713, 795, 797, 844
                 intuition/screens.h: 101
                 intuition/cghooks.h: 30
                 intuition/intuitionbase.h: 74
Window           pointer to void in struct Layer +0x0028 graphics/clip.h: *46
WindowPort       pointer to struct MsgPort in struct Window
         +0x005a intuition/intuition.h: *856
WorkBuffer       pointer to unsigned char in struct StringExtend
         +0x0010 intuition/sghooks.h: *28
WorkBuffer       pointer to unsigned char in struct SGWork
         +0x0008 intuition/sghooks.h: *37
WorkName         array [30] of unsigned char in struct Preferences
         +0x00ba intuition/preferences.h: *112
WriteChar        macro (1 argument)   dos/stdio.h: *18
```

Include File Cross Reference Page 142

```
WriteStr         macro (1 argument)   dos/stdio.h: *23
x                short int in struct tPoint +0x0000 graphics/gfx.h: *43
x                unsigned short int in struct SimpleSprite
         +0x0006 graphics/sprite.h: *25
xln_Init         pointer to function returning long int in struct ExtendedNode
         +0x0014 graphics/gfxnodes.h: *28
xln_Library      long int in struct ExtendedNode
         +0x0010 graphics/gfxnodes.h: *27
xln_Name         pointer to char in struct ExtendedNode
         +0x000a graphics/gfxnodes.h: *24
xln_Pred         pointer to struct Node in struct ExtendedNode
         +0x0004 graphics/gfxnodes.h: *21
xln_Pri          char in struct ExtendedNode +0x0009 graphics/gfxnodes.h: *23
xln_Subsystem    unsigned char in struct ExtendedNode
         +0x000e graphics/gfxnodes.h: *25
xln_Subtype      unsigned char in struct ExtendedNode
         +0x000f graphics/gfxnodes.h: *26
xln_Succ         pointer to struct Node in struct ExtendedNode
         +0x0000 graphics/gfxnodes.h: *20
xln_Type         unsigned char in struct ExtendedNode
         +0x0008 graphics/gfxnodes.h: *22
X                short int in struct (no tag)
         +0x0000 devices/inputevent.h: *91
X                unsigned short int in struct (no tag)
         +0x0000 devices/inputevent.h: *110
X                unsigned short int in struct (no tag)
         +0x0000 devices/inputevent.h: *114
X                unsigned short int in struct (no tag)
         +0x0000 intuition/screens.h: *71
X                short int in struct VSprite +0x0018 graphics/gels.h: *96
X                short int in struct (no tag)
         +0x0000 intuition/gadgetclass.h: *180
X                short int in struct (no tag)
         +0x0000 intuition/gadgetclass.h: *207
X                short int in struct (no tag)
         +0x0000 intuition/imageclass.h: *156
X                short int in struct (no tag)
         +0x0000 intuition/imageclass.h: *176
X                short int in struct (no tag)
         +0x0000 intuition/imageclass.h: *191
XAccel           short int in struct AnimOb +0x001a graphics/gels.h: *215
XOffset          char in struct Window +0x0050 intuition/intuition.h: *852
XOffset          char in struct Preferences
         +0x0064 intuition/preferences.h: *62
XTrans           short int in struct AnimComp +0x001c graphics/gels.h: *193
XVel             short int in struct AnimOb +0x0016 graphics/gels.h: *214
XY               pointer to short int in struct Border
         +0x0008 intuition/intuition.h: *604
y                short int in struct tPoint +0x0002 graphics/gfx.h: *43
y                unsigned short int in struct SimpleSprite
         +0x0008 graphics/sprite.h: *25
year             unsigned short int in struct ClockData
         +0x000a utility/date.h: *25
Y                short int in struct (no tag)
         +0x0002 devices/inputevent.h: *92
Y                unsigned short int in struct (no tag)
         +0x0002 devices/inputevent.h: *111
Y                unsigned short int in struct (no tag)
         +0x0002 devices/inputevent.h: *115
Y                unsigned short int in struct (no tag)
         +0x0002 intuition/screens.h: *72
Y                short int in struct VSprite +0x0016 graphics/gels.h: *96
Y                short int in struct (no tag)
         +0x0002 intuition/gadgetclass.h: *181
Y                short int in struct (no tag)
         +0x0002 intuition/gadgetclass.h: *208
```

```
Y                 short int in struct (no tag)
         +0x0002 intuition/imageclass.h: *157
Y                 short int in struct (no tag)
         +0x0002 intuition/imageclass.h: *177
Y                 short int in struct (no tag)
         +0x0002 intuition/imageclass.h: *192
YAccel            short int in struct AnimOb +0x0018 graphics/gels.h: *215
YOffset           char in struct Window +0x0051 intuition/intuition.h: *852
YOffset           char in struct Preferences
         +0x0065 intuition/preferences.h: *63
YTrans            short int in struct AnimComp +0x001a graphics/gels.h: *192
YVel              short int in struct AnimOb +0x0014 graphics/gels.h: *214
ZOOMED            #define WFLG_ZOOMED = 0x10000000  intuition/iobsolete.h: *171
ZOOMIMAGE         #define (0x01L) = 0x00000001  intuition/imageclass.h: *102
```

Cross Reference

Amiga Function Index

Page 1

--background--	532	(timer.device-2)
--background--	564	(FileSystem.resource-2)
AbleICR	557	(cia.resource-2)
AbortIO	127	(exec.library-3)
AbortIO	462	(audio.device-2)
AbortIO	498	(narrator.device-2)
AbortIO	523	(serial.device-2)
AbortPkt	48	(dos.library-4)
ActivateCxObj	21	(commodities.library-2)
ActivateGadget	304	(intuition.library-3)
ActivateWindow	304	(intuition.library-4)
ADCMD_ALLOCATE	464	(audio.device-5)
ADCMD_FINISH	465	(audio.device-7)
ADCMD_FREE	465	(audio.device-8)
ADCMD_LOCK	466	(audio.device-9)
ADCMD_PERVOL	466	(audio.device-10)
ADCMD_SETPREC	467	(audio.device-11)
ADCMD_WAITCYCLE	467	(audio.device-12)
AddAnimOb	206	(graphics.library-4)
AddAppIconA	455	(workbench.library-2)
AddAppMenuItemA	456	(workbench.library-4)
AddAppWindowA	457	(workbench.library-5)
AddBob	207	(graphics.library-5)
AddBootNode	182	(expansion.library-2)
AddBuffers	49	(dos.library-5)
AddClass	305	(intuition.library-5)
AddConfigDev	183	(expansion.library-3)
AddDevice	127	(exec.library-4)
AddDosEntry	49	(dos.library-6)
AddDosNode	183	(expansion.library-4)
AddFont	207	(graphics.library-6)
AddFreeList	273	(icon.library-2)
AddGadget	305	(intuition.library-6)
AddGList	306	(intuition.library-7)
AddHead	128	(exec.library-5)
AddICRVector	558	(cia.resource-3)
AddIEvents	22	(commodities.library-3)
AddIntServer	128	(exec.library-6)
AddLibrary	129	(exec.library-7)
AddMemList	129	(exec.library-8)
AddPart	50	(dos.library-7)
AddPort	130	(exec.library-9)
AddResource	130	(exec.library-10)
AddSegment	50	(dos.library-8)
AddSemaphore	131	(exec.library-11)
AddTail	131	(exec.library-12)
AddTask	132	(exec.library-13)
AddTime	533	(timer.device-3)
AddTOF	572	(amiga.lib.library-2)
AddVSprite	208	(graphics.library-7)
afp	577	(amiga.lib.library-11)
Alert	133	(exec.library-15)
AllocAbs	133	(exec.library-16)
AllocAslRequest	16	(asl.library-2)
Allocate	134	(exec.library-17)
AllocateTagItems	441	(utility.library-2)
AllocConfigDev	184	(expansion.library-6)
AllocDosObject	51	(dos.library-9)
AllocEntry	135	(exec.library-19)
AllocExpansionMem	185	(expansion.library-7)
AllocFileRequest	17	(asl.library-3)
AllocIFF	281	(iffparse.library-3)
AllocLocalItem	281	(iffparse.library-4)
AllocMem	135	(exec.library-20)
AllocMiscResource	565	(misc.resource-2)
AllocPotBits	567	(potgo.resource-2)

Amiga Function Index

Page 2

AllocRaster	208	(graphics.library-8)
AllocRemember	306	(intuition.library-8)
AllocSignal	136	(exec.library-22)
AllocTrap	137	(exec.library-23)
AllocUnit	560	(disk.resource-2)
AllocVec	137	(exec.library-24)
Amiga2Date	442	(utility.library-3)
AndRectRegion	209	(graphics.library-9)
AndRegionRegion	209	(graphics.library-10)
Animate	210	(graphics.library-11)
AreaCircle	210	(graphics.library-12)
AreaDraw	211	(graphics.library-13)
AreaEllipse	211	(graphics.library-14)
AreaEnd	212	(graphics.library-15)
AreaMove	212	(graphics.library-16)
ArgArrayDone	35	(cx_lib.library-2)
ArgArrayInit	36	(cx_lib.library-3)
ArgInt	37	(cx_lib.library-5)
ArgString	37	(cx_lib.library-6)
arnd	577	(amiga.lib.library-12)
AskFont	213	(graphics.library-17)
AskKeyMapDefault	375	(keymap.library-2)
AskSoftStyle	213	(graphics.library-18)
AslRequest	17	(asl.library-4)
AssignAdd	51	(dos.library-10)
AssignLate	52	(dos.library-11)
AssignLock	52	(dos.library-12)
AssignPath	53	(dos.library-13)
AttachCxObj	22	(commodities.library-4)
AttemptLockDosList	53	(dos.library-14)
AttemptLockLayerRom	214	(graphics.library-19)
AttemptSemaphore	138	(exec.library-25)
AutoRequest	307	(intuition.library-10)
AvailFonts	43	(diskfont.library-2)
AvailMem	138	(exec.library-26)
BeginIO	463	(audio.device-3)
BeginIO	524	(serial.device-3)
BeginIO	573	(amiga.lib.library-3)
BeginRefresh	308	(intuition.library-12)
BeginUpdate	378	(layers.library-2)
BehindLayer	379	(layers.library-3)
BitMapScale	214	(graphics.library-20)
BltBitMap	215	(graphics.library-21)
BltBitMapRastPort	216	(graphics.library-23)
BltClear	216	(graphics.library-24)
BltMaskBitMapRastPort	217	(graphics.library-25)
BltPattern	217	(graphics.library-26)
BltTemplate	218	(graphics.library-27)
BuildEasyRequestArgs	309	(intuition.library-14)
BuildSysRequest	310	(intuition.library-15)
BumpRevision	274	(icon.library-3)
CacheClearE	139	(exec.library-27)
CacheClearU	140	(exec.library-29)
CacheControl	140	(exec.library-30)
CallHookPkt	442	(utility.library-4)
Cause	141	(exec.library-31)
CBD_CHANGEHOOK	474	(clipboard.device-2)
CBD_CURRENTREADID	475	(clipboard.device-3)
CBD_CURRENTWRITEID	475	(clipboard.device-4)
CBD_POST	476	(clipboard.device-5)
CBump	218	(graphics.library-28)
CDInputHandler	480	(console.device-6)
CD_ASKDEFAULTKEYMAP	478	(console.device-2)
CD_ASKKEYMAP	479	(console.device-3)
CD_SETDEFAULTKEYMAP	479	(console.device-4)
CD_SETKEYMAP	480	(console.device-5)

Amiga Function Index

Page 3

CEND	219	(graphics.library-29)
ChangeMode	54	(dos.library-15)
ChangeSprite	219	(graphics.library-30)
ChangeWindowBox	311	(intuition.library-18)
CheckDate	443	(utility.library-5)
CheckIO	141	(exec.library-32)
CheckSignal	54	(dos.library-16)
CINIT	220	(graphics.library-31)
ClearCxObjError	23	(commodities.library-5)
ClearDMRequest	312	(intuition.library-19)
ClearEOL	220	(graphics.library-32)
ClearMenuStrip	312	(intuition.library-20)
ClearPointer	313	(intuition.library-21)
ClearRectRegion	221	(graphics.library-33)
ClearRegion	221	(graphics.library-34)
ClearRexxMsg	434	(rexxsyslib.library-2)
ClearScreen	222	(graphics.library-35)
Cli	55	(dos.library-17)
ClipBlit	222	(graphics.library-36)
CloneTagItems	443	(utility.library-6)
Close	55	(dos.library-18)
CloseClipboard	282	(iffparse.library-5)
CloseDevice	142	(exec.library-33)
CloseDevice	463	(audio.device-4)
CloseDevice	499	(narrator.device-3)
CloseDevice	524	(serial.device-4)
CloseFont	223	(graphics.library-37)
CloseIFF	282	(iffparse.library-6)
CloseLibrary	142	(exec.library-34)
CloseMonitor	223	(graphics.library-38)
CloseScreen	313	(intuition.library-22)
CloseWindow	314	(intuition.library-23)
CloseWorkBench	315	(intuition.library-25)
CMD_CLEAR	468	(audio.device-13)
CMD_CLEAR	481	(console.device-7)
CMD_CLEAR	494	(keyboard.device-2)
CMD_CLEAR	505	(parallel.device-2)
CMD_CLEAR	525	(serial.device-5)
CMD_CLEAR	538	(trackdisk.device-2)
CMD_FLUSH	468	(audio.device-14)
CMD_FLUSH	499	(narrator.device-4)
CMD_FLUSH	506	(parallel.device-3)
CMD_FLUSH	511	(printer.device-2)
CMD_FLUSH	525	(serial.device-6)
CMD_INVALID	512	(printer.device-3)
CMD_READ	469	(audio.device-15)
CMD_READ	476	(clipboard.device-6)
CMD_READ	481	(console.device-8)
CMD_Read	500	(narrator.device-5)
CMD_READ	506	(parallel.device-4)
CMD_READ	526	(serial.device-7)
CMD_READ	539	(trackdisk.device-3)
CMD_RESET	469	(audio.device-16)
CMD_RESET	501	(narrator.device-7)
CMD_RESET	507	(parallel.device-5)
CMD_RESET	512	(printer.device-4)
CMD_RESET	526	(serial.device-8)
CMD_START	470	(audio.device-17)
CMD_START	501	(narrator.device-8)
CMD_START	507	(parallel.device-6)
CMD_START	513	(printer.device-5)
CMD_START	527	(serial.device-9)
CMD_STOP	470	(audio.device-18)
CMD_STOP	502	(narrator.device-9)
CMD_STOP	508	(parallel.device-7)
CMD_STOP	513	(printer.device-6)

Amiga Function Index

Page 4

CMD_STOP	527	(serial.device-10)
CMD_UPDATE	471	(audio.device-19)
CMD_UPDATE	477	(clipboard.device-7)
CMD_UPDATE	539	(trackdisk.device-4)
CMD_WRITE	471	(audio.device-20)
CMD_WRITE	477	(clipboard.device-8)
CMD_WRITE	482	(console.device-9)
CMD_WRITE	502	(narrator.device-10)
CMD_WRITE	508	(parallel.device-8)
CMD_WRITE	514	(printer.device-7)
CMD_WRITE	528	(serial.device-11)
CMD_WRITE	540	(trackdisk.device-5)
CMOVE	224	(graphics.library-39)
CmpTime	533	(timer.device-4)
ColdReboot	143	(exec.library-35)
CollectionChunk	283	(iffparse.library-7)
CollectionChunks	283	(iffparse.library-8)
CompareDates	56	(dos.library-19)
ConfigBoard	185	(expansion.library-8)
CopyMem	143	(exec.library-36)
CopyMemQuick	144	(exec.library-37)
CopySBitMap	224	(graphics.library-40)
CreateArgstring	435	(rexxsyslib.library-3)
CreateBehindHookLayer	379	(layers.library-4)
CreateBehindLayer	380	(layers.library-5)
CreateContext	193	(gadtools.library-2)
CreateCxObj	23	(commodities.library-6)
CreateDir	56	(dos.library-20)
CreateExtIO	573	(amiga.lib.library-4)
CreateGadgetA	194	(gadtools.library-3)
CreateIORequest	144	(exec.library-38)
CreateMenusA	196	(gadtools.library-7)
CreateMsgPort	145	(exec.library-39)
CreateNewProc	57	(dos.library-21)
CreatePort	574	(amiga.lib.library-5)
CreateProc	57	(dos.library-22)
CreateRexxMsg	435	(rexxsyslib.library-4)
CreateTask	574	(amiga.lib.library-6)
CreateUpfrontHookLayer	380	(layers.library-6)
CreateUpfrontLayer	381	(layers.library-7)
CurrentChunk	284	(iffparse.library-9)
CurrentDir	58	(dos.library-23)
CurrentTime	315	(intuition.library-26)
CWAIT	225	(graphics.library-41)
CxBroker	24	(commodities.library-7)
CxCustom	38	(cx_lib.library-7)
CxDebug	38	(cx_lib.library-8)
CxFilter	39	(cx_lib.library-9)
CxMsgData	25	(commodities.library-9)
CxMsgID	25	(commodities.library-10)
CxMsgType	26	(commodities.library-11)
CxObjError	26	(commodities.library-12)
CxObjType	27	(commodities.library-13)
CxSender	39	(cx_lib.library-10)
CxSignal	40	(cx_lib.library-11)
CxTranslate	40	(cx_lib.library-12)
CxTypeFilter	41	(cx_lib.library-13)
Date2Amiga	444	(utility.library-7)
DateStamp	58	(dos.library-24)
DateToStr	59	(dos.library-25)
dbf	578	(amiga.lib.library-13)
Deallocate	145	(exec.library-40)
Debug	146	(exec.library-41)
Delay	60	(dos.library-27)
DeleteArgstring	436	(rexxsyslib.library-5)
DeleteCxObj	27	(commodities.library-14)

Amiga Function Index

DeleteCxObjAll	28	(commodities.library-15)
DeleteDiskObject	274	(icon.library-4)
DeleteExtIO	575	(amiga.lib.library-7)
DeleteFile	60	(dos.library-28)
DeleteIORequest	146	(exec.library-42)
DeleteLayer	381	(layers.library-8)
DeleteMsgPort	147	(exec.library-43)
DeletePort	575	(amiga.lib.library-8)
DeleteRexxMsg	436	(rexxsyslib.library-6)
DeleteTask	576	(amiga.lib.library-9)
DeleteVar	61	(dos.library-29)
DeviceProc	61	(dos.library-30)
Disable	147	(exec.library-44)
DisownBlitter	225	(graphics.library-42)
DisplayAlert	316	(intuition.library-27)
DisplayBeep	317	(intuition.library-29)
DisposeCxMsg	28	(commodities.library-16)
DisposeFontContents	44	(diskfont.library-4)
DisposeLayerInfo	382	(layers.library-9)
DisposeObject	317	(intuition.library-30)
DisposeRegion	226	(graphics.library-43)
DivertCxMsg	29	(commodities.library-17)
DoCollision	226	(graphics.library-44)
DoIO	148	(exec.library-45)
DoPkt	62	(dos.library-31)
DoubleClick	318	(intuition.library-31)
Draw	227	(graphics.library-45)
DrawBevelBoxA	197	(gadtools.library-9)
DrawBorder	318	(intuition.library-32)
DrawEllipse	227	(graphics.library-46)
DrawGList	228	(graphics.library-47)
DrawImage	319	(intuition.library-33)
DrawImageState	319	(intuition.library-34)
DupLock	62	(dos.library-32)
DupLockFromFH	63	(dos.library-33)
EasyRequestArgs	320	(intuition.library-35)
Enable	148	(exec.library-46)
EndNotify	63	(dos.library-34)
EndRefresh	321	(intuition.library-38)
EndRequest	322	(intuition.library-39)
EndUpdate	382	(layers.library-10)
Enqueue	149	(exec.library-47)
EnqueueCxObj	29	(commodities.library-18)
EntryHandler	284	(iffparse.library-10)
EraseImage	322	(intuition.library-40)
EraseRect	228	(graphics.library-48)
ErrorReport	64	(dos.library-35)
ExAll	64	(dos.library-36)
Examine	66	(dos.library-39)
ExamineFH	66	(dos.library-40)
Execute	67	(dos.library-41)
Exit	67	(dos.library-42)
ExitHandler	285	(iffparse.library-12)
ExNext	68	(dos.library-43)
Expunge	472	(audio.device-22)
ExtendFont	229	(graphics.library-49)
FastRand	576	(amiga.lib.library-10)
FattenLayerInfo	383	(layers.library-11)
Fault	68	(dos.library-44)
FGetC	69	(dos.library-45)
FGets	69	(dos.library-46)
FilePart	70	(dos.library-47)
FillRexxMsg	437	(rexxsyslib.library-7)
FilterTagChanges	444	(utility.library-8)
FilterTagItems	445	(utility.library-9)
FindArg	70	(dos.library-48)

Amiga Function Index

FindCliProc	71	(dos.library-49)
FindCollection	286	(iffparse.library-14)
FindConfigDev	186	(expansion.library-9)
FindDisplayInfo	229	(graphics.library-50)
FindDosEntry	71	(dos.library-50)
FindLocalItem	287	(iffparse.library-15)
FindName	149	(exec.library-48)
FindPort	150	(exec.library-49)
FindProp	287	(iffparse.library-16)
FindPropContext	288	(iffparse.library-17)
FindResident	150	(exec.library-50)
FindSegment	72	(dos.library-51)
FindSemaphore	151	(exec.library-51)
FindTagItem	445	(utility.library-10)
FindTask	151	(exec.library-52)
FindToolType	275	(icon.library-5)
FindVar	72	(dos.library-52)
Flood	230	(graphics.library-51)
Flush	73	(dos.library-53)
FontExtent	230	(graphics.library-52)
Forbid	152	(exec.library-53)
Format	73	(dos.library-54)
fpa	578	(amiga.lib.library-14)
fpbcd	579	(amiga.lib.library-15)
FPutC	74	(dos.library-55)
FPuts	74	(dos.library-56)
FRead	75	(dos.library-57)
FreeArgs	75	(dos.library-58)
FreeAslRequest	19	(asl.library-7)
FreeClass	323	(intuition.library-41)
FreeColorMap	231	(graphics.library-53)
FreeConfigDev	186	(expansion.library-10)
FreeCopList	231	(graphics.library-54)
FreeCprList	232	(graphics.library-55)
FreeDeviceProc	76	(dos.library-59)
FreeDiskObject	275	(icon.library-6)
FreeDosEntry	76	(dos.library-60)
FreeDosObject	77	(dos.library-61)
FreeEntry	152	(exec.library-54)
FreeExpansionMem	187	(expansion.library-11)
FreeFileRequest	19	(asl.library-8)
FreeFreeList	276	(icon.library-7)
FreeGadgets	197	(gadtools.library-10)
FreeGBuffers	232	(graphics.library-56)
FreeIEvents	41	(cx_lib.library-14)
FreeIFF	288	(iffparse.library-18)
FreeLocalItem	289	(iffparse.library-19)
FreeMem	153	(exec.library-55)
FreeMenus	198	(gadtools.library-11)
FreeMiscResource	566	(misc.resource-3)
FreePotBits	568	(potgo.resource-3)
FreeRaster	233	(graphics.library-57)
FreeScreenDrawInfo	323	(intuition.library-42)
FreeSignal	153	(exec.library-56)
FreeSprite	233	(graphics.library-58)
FreeSysRequest	324	(intuition.library-43)
FreeTagItems	446	(utility.library-11)
FreeTrap	154	(exec.library-57)
FreeUnit	561	(disk.resource-3)
FreeVec	154	(exec.library-58)
FreeVisualInfo	198	(gadtools.library-12)
FreeVPortCopLists	234	(graphics.library-59)
FWrite	77	(dos.library-62)
GadgetMouse	324	(intuition.library-44)
GetArgStr	78	(dos.library-63)
GetAttr	325	(intuition.library-45)

GetCC	155	(exec.library-59)
GetColorMap	234	(graphics.library-60)
GetConsoleTask	78	(dos.library-64)
GetCurrentBinding	187	(expansion.library-12)
GetCurrentDirName	79	(dos.library-65)
GetDefaultPubScreen	325	(intuition.library-46)
GetDefDiskObject	276	(icon.library-8)
GetDefPrefs	326	(intuition.library-47)
GetDeviceProc	79	(dos.library-66)
GetDiskObject	277	(icon.library-9)
GetDiskObjectNew	277	(icon.library-10)
GetDisplayInfoData	235	(graphics.library-61)
GetFileSysTask	80	(dos.library-67)
GetGBuffers	235	(graphics.library-62)
GetMsg	155	(exec.library-60)
GetPrefs	326	(intuition.library-48)
GetProgramDir	80	(dos.library-68)
GetProgramName	81	(dos.library-69)
GetPrompt	81	(dos.library-70)
GetRGB4	236	(graphics.library-63)
GetScreenData	327	(intuition.library-49)
GetScreenDrawInfo	328	(intuition.library-51)
GetSprite	236	(graphics.library-64)
GetSysTime	534	(timer.device-5)
GetTagData	446	(utility.library-12)
GetUnit	561	(disk.resource-4)
GetUnitID	562	(disk.resource-6)
GetVar	47	(dos.library-2)
GetVar	82	(dos.library-71)
GetVisualInfoA	199	(gadtools.library-13)
GetVPModeID	237	(graphics.library-65)
GfxAssociate	237	(graphics.library-66)
GfxFree	238	(graphics.library-67)
GfxLookUP	238	(graphics.library-68)
GfxNew	205	(graphics.library-2)
GfxNew	239	(graphics.library-69)
GiveUnit	563	(disk.resource-7)
GoodID	289	(iffparse.library-20)
GoodType	290	(iffparse.library-21)
GPD_ASKCTYPE	486	(gameport.device-2)
GPD_ASKTRIGGER	487	(gameport.device-3)
GPD_READEVENT	487	(gameport.device-4)
GPD_SETCTYPE	488	(gameport.device-5)
GPD_SETTRIGGER	488	(gameport.device-6)
GT_BeginRefresh	199	(gadtools.library-14)
GT_EndRefresh	200	(gadtools.library-15)
GT_FilterIMsg	200	(gadtools.library-16)
GT_GetIMsg	201	(gadtools.library-17)
GT_PostFilterIMsg	201	(gadtools.library-18)
GT_RefreshWindow	202	(gadtools.library-19)
GT_ReplyIMsg	202	(gadtools.library-20)
GT_SetGadgetAttrsA	203	(gadtools.library-21)
HotKey	42	(cx_lib.library-15)
IDtoStr	290	(iffparse.library-22)
IEEEDPAbs	393	(mathieeedoubbas.library-2)
IEEEDPAcos	400	(mathieeedoubtrans.library-2)
IEEEDPAdd	394	(mathieeedoubbas.library-3)
IEEEDPAsin	401	(mathieeedoubtrans.library-3)
IEEEDPAtan	401	(mathieeedoubtrans.library-4)
IEEEDPCeil	394	(mathieeedoubbas.library-4)
IEEEDPCmp	395	(mathieeedoubbas.library-5)
IEEEDPCos	402	(mathieeedoubtrans.library-5)
IEEEDPCosh	402	(mathieeedoubtrans.library-6)
IEEEDPDiv	395	(mathieeedoubbas.library-6)
IEEEDPExp	403	(mathieeedoubtrans.library-7)
IEEEDPFieee	403	(mathieeedoubtrans.library-8)

IEEEDPFix	396	(mathieeedoubbas.library-7)
IEEEDPFloor	396	(mathieeedoubbas.library-8)
IEEEDPFlt	397	(mathieeedoubbas.library-9)
IEEEDPLog	404	(mathieeedoubtrans.library-9)
IEEEDPLog10	404	(mathieeedoubtrans.library-10)
IEEEDPMul	397	(mathieeedoubbas.library-10)
IEEEDPNeg	398	(mathieeedoubbas.library-11)
IEEEDPPow	405	(mathieeedoubtrans.library-11)
IEEEDPSin	405	(mathieeedoubtrans.library-12)
IEEEDPSincos	406	(mathieeedoubtrans.library-13)
IEEEDPSinh	406	(mathieeedoubtrans.library-14)
IEEEDPSqrt	407	(mathieeedoubtrans.library-15)
IEEEDPSub	398	(mathieeedoubbas.library-12)
IEEEDPTan	407	(mathieeedoubtrans.library-16)
IEEEDPTanh	408	(mathieeedoubtrans.library-17)
IEEEDPTieee	408	(mathieeedoubtrans.library-18)
IEEEDPTst	399	(mathieeedoubbas.library-13)
IEEESPAbs	409	(matheieeesingbas.library-2)
IEEESPAcos	416	(matheieeesingtrans.library-2)
IEEESPAdd	410	(matheieeesingbas.library-3)
IEEESPAsin	417	(matheieeesingtrans.library-3)
IEEESPAtan	417	(matheieeesingtrans.library-4)
IEEESPCeil	410	(matheieeesingbas.library-4)
IEEESPCmp	411	(matheieeesingbas.library-5)
IEEESPCos	418	(matheieeesingtrans.library-5)
IEEESPCosh	418	(matheieeesingtrans.library-6)
IEEESPDiv	411	(matheieeesingbas.library-6)
IEEESPExp	419	(matheieeesingtrans.library-7)
IEEESPFieee	419	(matheieeesingtrans.library-8)
IEEESPFix	412	(matheieeesingbas.library-7)
IEEESPFloor	412	(matheieeesingbas.library-8)
IEEESPFlt	413	(matheieeesingbas.library-9)
IEEESPLog	420	(matheieeesingtrans.library-9)
IEEESPLog10	420	(matheieeesingtrans.library-10)
IEEESPMul	413	(matheieeesingbas.library-10)
IEEESPNeg	414	(matheieeesingbas.library-11)
IEEESPPow	421	(matheieeesingtrans.library-11)
IEEESPSin	421	(matheieeesingtrans.library-12)
IEEESPSincos	422	(matheieeesingtrans.library-13)
IEEESPSinh	422	(matheieeesingtrans.library-14)
IEEESPSqrt	423	(matheieeesingtrans.library-15)
IEEESPSub	414	(matheieeesingbas.library-12)
IEEESPTan	423	(matheieeesingtrans.library-16)
IEEESPTanh	424	(matheieeesingtrans.library-17)
IEEESPTieee	424	(matheieeesingtrans.library-18)
IEEESPTst	415	(matheieeesingbas.library-13)
IND_ADDHANDLER	489	(input.device-2)
IND_REMHANDLER	490	(input.device-3)
IND_SETMPORT	490	(input.device-4)
IND_SETMTRIG	491	(input.device-5)
IND_SETMTYPE	491	(input.device-6)
IND_SETPERIOD	492	(input.device-7)
IND_SETTHRESH	492	(input.device-8)
IND_WRITEEVENT	493	(input.device-9)
Info	82	(dos.library-72)
Inhibit	83	(dos.library-73)
InitArea	239	(graphics.library-70)
InitBitMap	240	(graphics.library-71)
InitCode	156	(exec.library-61)
InitGels	240	(graphics.library-72)
InitGMasks	241	(graphics.library-73)
InitIFF	291	(iffparse.library-23)
InitIFFasClip	292	(iffparse.library-25)
InitIFFasDOS	292	(iffparse.library-26)
InitLayers	383	(layers.library-12)
InitMasks	241	(graphics.library-74)

Amiga Function Index

Page 9

InitRastPort	242	(graphics.library-75)
InitRequester	328	(intuition.library-52)
InitResident	156	(exec.library-62)
InitSemaphore	157	(exec.library-63)
InitStruct	157	(exec.library-64)
InitTmpRas	242	(graphics.library-76)
InitView	243	(graphics.library-77)
InitVPort	243	(graphics.library-78)
Input	83	(dos.library-74)
InsertCxObj	30	(commodities.library-19)
InstallClipRegion	384	(layers.library-13)
InstallLayerHook	384	(layers.library-14)
InternalLoadSeg	84	(dos.library-75)
InternalUnLoadSeg	84	(dos.library-76)
IntuiTextLength	329	(intuition.library-53)
InvertKeyMap	30	(commodities.library-20)
InvertString	42	(cx_lib.library-16)
IoErr	85	(dos.library-77)
IsFileSystem	85	(dos.library-78)
IsInteractive	86	(dos.library-79)
IsRexxMsg	437	(rexxsyslib.library-8)
ItemAddress	329	(intuition.library-54)
KBD_ADDRESETHANDLER	495	(keyboard.device-3)
KBD_READEVENT	495	(keyboard.device-4)
KBD_READMATRIX	496	(keyboard.device-5)
KBD_REMRESETHANDLER	496	(keyboard.device-6)
KBD_RESETHANDLERDONE	497	(keyboard.device-7)
KCmpStr	583	(debug.library-2)
KGetChar	584	(debug.library-3)
KGetNum	584	(debug.library-4)
KMayGetChar	585	(debug.library-5)
KPrintF	585	(debug.library-6)
KPutChar	586	(debug.library-7)
KPutStr	586	(debug.library-8)
LayoutMenuItemsA	204	(gadtools.library-23)
LayoutMenusA	204	(gadtools.library-24)
LengthArgstring	438	(rexxsyslib.library-9)
LoadRGB4	244	(graphics.library-79)
LoadSeg	86	(dos.library-80)
LoadView	244	(graphics.library-80)
LocalItemData	293	(iffparse.library-27)
Lock	87	(dos.library-81)
LockDosList	87	(dos.library-82)
LockIBase	330	(intuition.library-55)
LockLayer	385	(layers.library-15)
LockLayerInfo	385	(layers.library-16)
LockLayerRom	245	(graphics.library-81)
LockLayers	386	(layers.library-17)
LockPubScreen	330	(intuition.library-56)
LockPubScreenList	331	(intuition.library-57)
LockRecord	88	(dos.library-83)
LockRecords	88	(dos.library-84)
LockRexxBase	438	(rexxsyslib.library-10)
MakeClass	331	(intuition.library-58)
MakeDosEntry	89	(dos.library-85)
MakeDosNode	188	(expansion.library-13)
MakeFunctions	158	(exec.library-65)
MakeLibrary	158	(exec.library-66)
MakeLink	89	(dos.library-86)
MakeScreen	332	(intuition.library-60)
MakeVPort	245	(graphics.library-82)
MapANSI	376	(keymap.library-3)
MapRawKey	377	(keymap.library-5)
MapTags	447	(utility.library-13)
MatchEnd	90	(dos.library-87)
MatchFirst	90	(dos.library-88)

Amiga Function Index

Page 10

MatchNext	91	(dos.library-89)
MatchPattern	91	(dos.library-90)
MatchPatternNoCase	92	(dos.library-91)
MatchToolValue	278	(icon.library-11)
MaxCli	92	(dos.library-92)
ModeNotAvailable	246	(graphics.library-83)
ModifyIDCMP	333	(intuition.library-61)
ModifyProp	334	(intuition.library-63)
Move	246	(graphics.library-84)
MoveLayer	386	(layers.library-18)
MoveLayerInFrontOf	387	(layers.library-19)
MoveScreen	334	(intuition.library-64)
MoveSizeLayer	387	(layers.library-20)
MoveSprite	247	(graphics.library-85)
MoveWindow	335	(intuition.library-65)
MoveWindowInFrontOf	335	(intuition.library-66)
MrgCop	247	(graphics.library-86)
NameFromFH	93	(dos.library-93)
NameFromLock	93	(dos.library-94)
NewFontContents	45	(diskfont.library-5)
NewLayerInfo	388	(layers.library-21)
NewList	132	(exec.library-14)
NewList	579	(amiga.lib.library-16)
NewLoadSeg	94	(dos.library-95)
NewModifyProp	336	(intuition.library-67)
NewObject	336	(intuition.library-68)
NewRegion	248	(graphics.library-87)
NewScaledDiskFont	45	(diskfont.library-6)
NextDisplayInfo	248	(graphics.library-88)
NextDosEntry	94	(dos.library-96)
NextObject	337	(intuition.library-69)
NextPubScreen	337	(intuition.library-70)
NextTagItem	448	(utility.library-15)
ObtainBattSemaphore	554	(battmem.resource-2)
ObtainConfigBinding	189	(expansion.library-15)
ObtainGIRPort	338	(intuition.library-71)
ObtainSemaphore	159	(exec.library-67)
ObtainSemaphoreList	159	(exec.library-68)
ObtainSemaphoreShared	160	(exec.library-69)
OffGadget	338	(intuition.library-72)
OffMenu	339	(intuition.library-73)
OldOpenLibrary	126	(exec.library-2)
OldOpenLibrary	160	(exec.library-70)
OnGadget	339	(intuition.library-74)
OnMenu	340	(intuition.library-75)
Open	95	(dos.library-97)
OpenClipboard	293	(iffparse.library-28)
OpenDevice	161	(exec.library-71)
OpenDevice	473	(audio.device-23)
OpenDevice	483	(console.device-12)
OpenDevice	503	(narrator.device-12)
OpenDevice	509	(parallel.device-9)
OpenDevice	528	(serial.device-12)
OpenDiskFont	46	(diskfont.library-7)
OpenFont	249	(graphics.library-89)
OpenFromLock	95	(dos.library-98)
OpenIFF	294	(iffparse.library-29)
OpenLibrary	161	(exec.library-72)
OpenMonitor	249	(graphics.library-90)
OpenResource	162	(exec.library-73)
OpenScreen	340	(intuition.library-76)
OpenScreenTagList	343	(intuition.library-82)
OpenWindow	303	(intuition.library-2)
OpenWindow	344	(intuition.library-84)
OpenWindowTagList	350	(intuition.library-96)
OpenWorkBench	351	(intuition.library-97)

Amiga Function Index

OrRectRegion	250	(graphics.library-91)
OrRegionRegion	250	(graphics.library-92)
Output	96	(dos.library-99)
OwnBlitter	251	(graphics.library-93)
PackBoolTags	448	(utility.library-16)
ParentChunk	294	(iffparse.library-30)
ParentDir	96	(dos.library-100)
ParentOfFH	97	(dos.library-101)
ParseIFF	295	(iffparse.library-31)
ParseIX	31	(commodities.library-21)
ParsePattern	97	(dos.library-102)
ParsePatternNoCase	98	(dos.library-103)
PathPart	98	(dos.library-104)
PDCMD_QUERY	509	(parallel.device-10)
PDCMD_SETPARAMS	510	(parallel.device-11)
PeekQualifier	493	(input.device-10)
Permit	162	(exec.library-74)
PointInImage	351	(intuition.library-98)
PolyDraw	251	(graphics.library-94)
PopChunk	296	(iffparse.library-33)
PRD_DUMPRPORT	515	(printer.device-10)
PRD_PRTCOMMAND	521	(printer.device-21)
PRD_QUERY	521	(printer.device-22)
PRD_RAWWRITE	522	(printer.device-23)
printf	580	(amiga.lib.library-17)
PrintFault	99	(dos.library-105)
PrintIText	352	(intuition.library-99)
Procure	163	(exec.library-75)
PropChunk	296	(iffparse.library-34)
PropChunks	297	(iffparse.library-35)
PubScreenStatus	352	(intuition.library-100)
PushChunk	297	(iffparse.library-36)
PutDefDiskObject	278	(icon.library-12)
PutDiskObject	279	(icon.library-13)
PutMsg	163	(exec.library-76)
PutStr	99	(dos.library-106)
PWrite	522	(printer.device-24)
QBlit	252	(graphics.library-95)
QBSBlit	252	(graphics.library-96)
QueryOverscan	353	(intuition.library-101)
RangeRand	581	(amiga.lib.library-19)
RawDoFmt	164	(exec.library-77)
RawKeyConvert	485	(console.device-15)
Read	100	(dos.library-107)
ReadArgs	100	(dos.library-108)
ReadBattClock	552	(battclock.resource-2)
ReadBattMem	555	(battmem.resource-3)
ReadChunkBytes	298	(iffparse.library-37)
ReadChunkRecords	298	(iffparse.library-38)
ReadEClock	534	(timer.device-6)
ReadExpansionByte	189	(expansion.library-16)
ReadExpansionRom	190	(expansion.library-17)
ReadItem	101	(dos.library-110)
ReadLink	102	(dos.library-111)
ReadPixel	253	(graphics.library-97)
ReadPixelArray8	253	(graphics.library-98)
ReadPixelLine8	254	(graphics.library-99)
ReadUnitID	563	(disk.resource-8)
RectFill	254	(graphics.library-100)
RefreshGadgets	353	(intuition.library-102)
RefreshGList	354	(intuition.library-104)
RefreshTagItemClones	449	(utility.library-18)
RefreshWindowFrame	355	(intuition.library-105)
Relabel	102	(dos.library-112)
ReleaseBattSemaphore	555	(battmem.resource-4)
ReleaseConfigBinding	190	(expansion.library-18)

Amiga Function Index

ReleaseGIRPort	355	(intuition.library-106)
ReleaseSemaphore	165	(exec.library-79)
ReleaseSemaphoreList	165	(exec.library-80)
RemakeDisplay	356	(intuition.library-107)
RemAssignList	103	(dos.library-113)
RemBob	255	(graphics.library-101)
RemConfigDev	191	(expansion.library-19)
RemDevice	166	(exec.library-81)
RemDosEntry	103	(dos.library-114)
RemFont	255	(graphics.library-102)
RemHead	166	(exec.library-82)
RemIBob	256	(graphics.library-103)
RemICRVector	559	(cia.resource-5)
RemIntServer	167	(exec.library-83)
RemLibrary	167	(exec.library-84)
Remove	168	(exec.library-85)
RemoveAppIcon	457	(workbench.library-6)
RemoveAppMenuItem	458	(workbench.library-7)
RemoveAppWindow	458	(workbench.library-8)
RemoveClass	356	(intuition.library-108)
RemoveCxObj	31	(commodities.library-22)
RemoveGadget	357	(intuition.library-109)
RemoveGList	357	(intuition.library-110)
RemPort	168	(exec.library-86)
RemResource	169	(exec.library-87)
RemSegment	104	(dos.library-115)
RemSemaphore	169	(exec.library-88)
RemTail	170	(exec.library-89)
RemTask	170	(exec.library-90)
RemTOF	581	(amiga.lib.library-20)
RemVSprite	256	(graphics.library-104)
Rename	104	(dos.library-116)
ReplyMsg	171	(exec.library-91)
ReplyPkt	105	(dos.library-117)
ReportMouse	358	(intuition.library-111)
Request	359	(intuition.library-113)
RequestFile	20	(asl.library-9)
ResetBattClock	553	(battclock.resource-3)
ResetMenuStrip	359	(intuition.library-114)
RethinkDisplay	360	(intuition.library-115)
RouteCxMsg	32	(commodities.library-23)
RunCommand	105	(dos.library-118)
SameDevice	106	(dos.library-119)
SameLock	106	(dos.library-120)
ScalerDiv	257	(graphics.library-105)
ScreenToBack	360	(intuition.library-116)
ScreenToFront	361	(intuition.library-117)
ScrollLayer	388	(layers.library-22)
ScrollRaster	257	(graphics.library-106)
ScrollVPort	258	(graphics.library-107)
SDCMD_BREAK	529	(serial.device-14)
SDCMD_QUERY	530	(serial.device-15)
SDCMD_SETPARAMS	530	(serial.device-16)
SDivMod32	450	(utility.library-19)
Seek	107	(dos.library-121)
SelectInput	107	(dos.library-122)
SelectOutput	108	(dos.library-123)
SendIO	171	(exec.library-92)
SendPkt	108	(dos.library-124)
SetAPen	258	(graphics.library-108)
SetArgStr	109	(dos.library-125)
SetAttrsA	361	(intuition.library-118)
SetBPen	259	(graphics.library-109)
SetCollision	259	(graphics.library-110)
SetComment	109	(dos.library-126)
SetConsoleTask	110	(dos.library-127)

Amiga Function Index

SetCurrentBinding	191	(expansion.library-20)
SetCurrentDirName	110	(dos.library-128)
SetCxObjPri	32	(commodities.library-24)
SetDefaultPubScreen	362	(intuition.library-119)
SetDMRequest	362	(intuition.library-120)
SetDrMd	260	(graphics.library-111)
SetEditHook	363	(intuition.library-121)
SetExcept	172	(exec.library-93)
SetFileDate	111	(dos.library-129)
SetFileSize	111	(dos.library-130)
SetFileSysTask	112	(dos.library-131)
SetFilter	33	(commodities.library-25)
SetFilterIX	33	(commodities.library-26)
SetFont	260	(graphics.library-112)
SetFunction	172	(exec.library-94)
SetGadgetAttrsA	363	(intuition.library-122)
SetICR	559	(cia.resource-6)
SetIntVector	173	(exec.library-95)
SetIoErr	112	(dos.library-132)
SetLocalItemPurge	299	(iffparse.library-39)
SetMenuStrip	364	(intuition.library-123)
SetMode	113	(dos.library-133)
SetMouseQueue	364	(intuition.library-124)
SetOPen	261	(graphics.library-113)
SetPointer	365	(intuition.library-125)
SetPrefs	365	(intuition.library-126)
SetProgramDir	113	(dos.library-134)
SetProgramName	114	(dos.library-135)
SetPrompt	114	(dos.library-136)
SetProtection	115	(dos.library-137)
SetPubScreenModes	366	(intuition.library-127)
SetRast	261	(graphics.library-114)
SetRGB4	262	(graphics.library-115)
SetRGB4CM	262	(graphics.library-116)
SetSignal	173	(exec.library-96)
SetSoftStyle	263	(graphics.library-117)
SetSR	174	(exec.library-97)
SetTaskPri	174	(exec.library-98)
SetTranslate	34	(commodities.library-27)
SetVar	115	(dos.library-138)
SetVar	48	(dos.library-3)
SetVBuf	116	(dos.library-139)
SetWindowTitles	366	(intuition.library-128)
ShowTitle	367	(intuition.library-129)
Signal	175	(exec.library-99)
SizeLayer	389	(layers.library-23)
SizeWindow	367	(intuition.library-130)
SMult32	450	(utility.library-20)
SortGList	263	(graphics.library-118)
SPAcos	425	(mathtrans.library-2)
SPAsin	426	(mathtrans.library-3)
SPAtan	426	(mathtrans.library-4)
SPCos	427	(mathtrans.library-5)
SPCosh	427	(mathtrans.library-6)
SPExp	428	(mathtrans.library-7)
SPFieee	428	(mathtrans.library-8)
SplitName	116	(dos.library-140)
SPLog	429	(mathtrans.library-9)
SPLog10	429	(mathtrans.library-10)
SPPow	430	(mathtrans.library-11)
sprintf	582	(amiga.lib.library-21)
SPSin	430	(mathtrans.library-12)
SPSincos	431	(mathtrans.library-13)
SPSinh	431	(mathtrans.library-14)
SPSqrt	432	(mathtrans.library-15)
SPTan	432	(mathtrans.library-16)

Amiga Function Index

SPTanh	433	(mathtrans.library-17)
SPTieee	433	(mathtrans.library-18)
StartNotify	117	(dos.library-141)
stdio	582	(amiga.lib.library-22)
StopChunk	299	(iffparse.library-40)
StopChunks	300	(iffparse.library-41)
StopOnExit	300	(iffparse.library-42)
StoreItemInContext	301	(iffparse.library-43)
StoreLocalItem	301	(iffparse.library-44)
Stricmp	451	(utility.library-21)
StripFont	264	(graphics.library-119)
Strnicmp	451	(utility.library-22)
StrToDate	117	(dos.library-142)
StrToLong	118	(dos.library-144)
SubTime	535	(timer.device-7)
SumKickData	175	(exec.library-100)
SumLibrary	176	(exec.library-102)
SuperState	177	(exec.library-103)
Supervisor	177	(exec.library-104)
SwapBitsRastPortClipRect	389	(layers.library-24)
SyncSBitMap	264	(graphics.library-120)
SysReqHandler	368	(intuition.library-131)
SystemTagList	119	(dos.library-145)
TagInArray	452	(utility.library-23)
TD_ADDCHANGEINT	540	(trackdisk.device-6)
TD_CHANGENUM	541	(trackdisk.device-7)
TD_CHANGESTATE	541	(trackdisk.device-8)
TD_EJECT	542	(trackdisk.device-9)
TD_FORMAT	542	(trackdisk.device-10)
TD_GETDRIVETYPE	543	(trackdisk.device-11)
TD_GETGEOMETRY	543	(trackdisk.device-12)
TD_GETNUMTRACKS	544	(trackdisk.device-13)
TD_MOTOR	544	(trackdisk.device-14)
TD_PROTSTATUS	545	(trackdisk.device-15)
TD_RAWREAD	545	(trackdisk.device-16)
TD_RAWWRITE	546	(trackdisk.device-17)
TD_REMCHANGEINT	546	(trackdisk.device-18)
TD_SEEK	547	(trackdisk.device-19)
Text	265	(graphics.library-121)
TextExtent	265	(graphics.library-122)
TextFit	266	(graphics.library-123)
TextLength	266	(graphics.library-124)
ThinLayerInfo	390	(layers.library-25)
ToLower	452	(utility.library-24)
ToUpper	453	(utility.library-25)
Translate	440	(translator.library-2)
TR_ADDREQUEST	535	(timer.device-8)
TR_GETSYSTIME	536	(timer.device-9)
TR_SETSYSTIME	536	(timer.device-10)
TypeOfMem	178	(exec.library-105)
UDivMod32	453	(utility.library-26)
UMult32	454	(utility.library-27)
UnGetC	119	(dos.library-146)
UnLoadSeg	120	(dos.library-147)
UnLock	120	(dos.library-148)
UnLockDosList	121	(dos.library-149)
UnlockIBase	369	(intuition.library-133)
UnlockLayer	390	(layers.library-26)
UnlockLayerInfo	391	(layers.library-27)
UnlockLayerRom	267	(graphics.library-125)
UnlockLayers	391	(layers.library-28)
UnlockPubScreen	369	(intuition.library-134)
UnlockPubScreenList	370	(intuition.library-135)
UnLockRecord	121	(dos.library-150)
UnLockRecords	122	(dos.library-151)
UnlockRexxBase	439	(rexxsyslib.library-11)

UpfrontLayer 392 (layers.library-29)
UserState 178 (exec.library-106)
Vacate 179 (exec.library-107)
VBeamPos 267 (graphics.library-126)
VFPrintf 122 (dos.library-152)
VFWritef 123 (dos.library-153)
VideoControl 268 (graphics.library-127)
ViewAddress 370 (intuition.library-136)
ViewPortAddress 371 (intuition.library-137)
VPrintf 123 (dos.library-154)
Wait 179 (exec.library-108)
WaitBlit 268 (graphics.library-128)
WaitBOVP 269 (graphics.library-129)
WaitForChar 124 (dos.library-155)
WaitIO 180 (exec.library-109)
WaitPkt 124 (dos.library-156)
WaitPort 180 (exec.library-110)
WaitTOF 269 (graphics.library-130)
WBenchToBack 371 (intuition.library-138)
WBenchToFront 372 (intuition.library-139)
WeighTAMatch 270 (graphics.library-131)
WhichLayer 392 (layers.library-30)
WindowLimits 372 (intuition.library-140)
WindowToBack 373 (intuition.library-141)
WindowToFront 373 (intuition.library-142)
Write 125 (dos.library-157)
WriteBattClock 553 (battclock.resource-4)
WriteBattMem 556 (battmem.resource-5)
WriteChars 125 (dos.library-158)
WriteChunkBytes 302 (iffparse.library-45)
WriteChunkRecords 302 (iffparse.library-46)
WriteExpansionByte 192 (expansion.library-21)
WritePixel 270 (graphics.library-132)
WritePixelArray8 271 (graphics.library-133)
WritePixelLine8 271 (graphics.library-134)
WritePotgo 568 (potgo.resource-4)
XorRectRegion 206 (graphics.library-3)
XorRectRegion 272 (graphics.library-135)
XorRegionRegion 272 (graphics.library-136)
ZipWindow 374 (intuition.library-143)

Amiga Programming

>$38.95 USA
>$50.95 CANADA

THIRD EDITION

AMIGA TECHNICAL REFERENCE SERIES

AMIGA® ROM Kernel Reference Manual

INCLUDES AND AUTODOCS

The Amiga computers are exciting high-performance microcomputers with superb graphics, sound, multiwindow and multitasking capabilities. Their technologically advanced hardware is designed around the Motorola 68000 microprocessor family and sophisticated custom chips. The Amiga's unique system software provides programmers with unparalleled power, flexibility, and convenience in designing and creating programs.

This manual is the definitive source of information on the routines and data structures in the Amiga's operating system. This new edition has been updated to include the hundreds of new system functions and structures in Release 2. It includes:

- Function descriptions for each Library, Device, and Resource call in the system
- Complete listings of the Amiga's C and Assembly language include files
- Documentation and source code for functions in the amiga.lib linker library
- Cross-reference charts to help you find the Amiga information you need

Written by the technical experts at Commodore-Amiga, Inc., the ***Amiga ROM Kernel Reference Manual: Includes and Autodocs*** is an essential reference tool for all Amiga programmers who want to take full advantage of the Amiga's impressive capabilities.

The new AMIGA TECHNICAL REFERENCE SERIES has been revised and updated to provide a comprehensive reference and tutorial for the entire line of Amiga computers and for Release 2 of the operating system. Other titles in the series include:

Amiga User Interface Style Guide
Amiga ROM Kernel Reference Manual: Devices
Amiga ROM Kernel Reference Manual: Libraries
Amiga Hardware Reference Manual

53895>
9 780201 567731
ISBN 0-201-56773-3
56773

Cover design by Hannus Design Associates

Addison-Wesley Publishing Company, Inc.

This was brought to you

from the archives of

http://retro-commodore.eu